(2)

# The Perception of Articulated Motion: Recognizing Moving Light Displays

Nigel H. Goddard



Technical Report 405
June 1992



UNIVERSITY OF
ROCHESTER
COMPUTER SCIENCE

# The Perception of Articulated Motion: Recognizing Moving Light Displays

by

Nigel H. Goddard

Submitted in Partial Fulfillment

of the

Requirements for the Degree

DOCTOR OF PHILOSOPHY

Supervised by Jerome A. Feldman

Department of Computer Science

University of Rochester

Rochester, New York

April 1992

# Curriculum Vitae

Nigel Goddard was born in 1958 in Lichfield, England. He received a B.A from Cambridge University in 1979 and an MA in 1984, having studied Mathematics, Engineering and Management Sciences. He spent the intervening years teaching, developing compilers at Manchester University and Phillips Telecommunicatie NV in Hilversum, and programming a rule-based system to assist knowledge-engineers at the Cognitive Studies Institute, University of Amsterdam.

In the autumn of 1985 he entered the Ph.D. program in the Computer Science Department at the University of Rochester. Almost immediately he began working with Jerome Feldman on connectionist vision. In the summer of 1986, he worked at the Information Sciences Institute in Los Angeles, on a text-generator named, of all names, Nigel. He returned to Rochester and begin his thesis research with a study of the scientific data concerning Johansson's Moving Light Displays. He also took over responsibility for the Rochester Connectionist Simulator, adding new features and preparing it for release as a research tool to the research community. In the summer of 1987, the simulator was released and he went to work for Hughes Research Laboratories in Malibu, where he continued his thesis research. During the 1987/1988 academic year he spent half his time in the Computational Neurosciences Laboratory at the California Institute of Technology, learning about the complexity and diversity of real neural networks. While at Hughes, he worked mainly on the problem of modeling human recognition of Moving Light Displays, but also upgraded the Rochester Connectionist Simulator twice. At the end of 1989, he moved to Pittsburgh and completed his thesis work at Carnegie Mellon University. In February 1992 he joined the staff of the Pittsburgh Supercomputer Center as a Scientific Specialist in the Biomedical program.

# Acknowledgments

Many people have helped me along the way in the production of this thesis, but none more so than my advisor, Jerome Feldman. Without his inspiration, guidance and continuing support, this thesis would not have been undertaken and completed. The overall framework within which this research lies is his, and many of the ideas embodied in this document are due to him or discussions with him. His straightforward, incisive criticism has been invaluable. He has shown me what it means to conduct research and to be a scientist. I would also like to thank him for his understanding and tolerance of my peripatetic behavior, and his unfailing ability to provide support when it was most needed.

My thesis committee members have been exceptionally helpful in different ways. Randal Nelson and Rob Fowler were instrumental in persuading me to conduct the second set of experiments and re-examine the role of what eventually became the attention mechanism. Randal kept reminding me to consider the reality of clutter and occlusion in real world environments, and Rob insisted I articulate and generalize the lessons I had learned. John Maunsell provided invaluable advice and knowledge of the neurophysiological data and current theories, and countered my tendency to reduce brain mechanisms to simple computational devices. I would particularly like to thank him for the care with which he examined drafts and the understanding he showed for my situation as a student *in absentia*. Mary Hayhoe kept bringing me back to the amazing human perceptual abilities, and the equally fascinating failures in achieving veridical perception.

At the University of Rochester, I would like to thank James Allen and Tom Leblanc in their roles as consecutive Chairman of the Computer Science Department. Both were tolerant of my studying *in absentia*, and Tom's intervention was critical in ensuring I stayed on track in the final stages. Peggy Frantz helped enormously in navigating the paperwork and other tasks made difficult by my absence from the Department. Liudy Bukys was invaluable in organizing the release and support of the Rochester Connectionist Simulator.

Fellow students at Rochester helped me adjust to life in a foreign country, and some provided immensely useful comments on this dissertation, particularly: Paul Cooper, who always sees the big picture as well as the detail; Tom Olson, whose dissertation was an essential foundation this work; Susan Weber, who commented on an early draft in great detail and with considerable editorial skill; and Mark Fanty, who had many insightful comments to make on simulator design.

My colleagues at Hughes Research Laboratories deserve thanks for their toleration of my single-mindedness in pursuing this line of research, despite various pressures to do otherwise. I owe a great debt to Charlie Dolan, for his professional and personal support during my time at Hughes and since then.

Without Christof Koch's support, I would not have had the opportunity to learn about the wonderful world of neuroscience in such detail. The time I spent in his lab at Caltech was invaluable in showing me how simple our computational models are compared to the intricacies of real neural systems.

I was unusually fortunate in being given access to lab space and computers at Carnegie Mellon, initially by Dave Touretzky and later by Takeo Kanade. Without their help, completing work on this thesis would have been impossible. Interacting with the Boltzmann research group and the Image Understanding research group enriched my perspective. Jay McClelland and his PDP research group provided an important forum for discussion of connectionist cognitive modeling.

The second set of experiments would have been impossible without the help of Ray Burdett of the Department of Physical Therapy at the University of Pittsburgh, who provided the equipment and expertise used to collect the data samples of different gaits.

The simulation software would have been much more difficult to code and debug were it not for the programming environment provided by the Free Software Foundation's GNU project, in particular the compiler G++ and symbolic debugger GDB. Access to source code for the compiler and debugger was no less than critical. I thank all the contributors to the GNU project.

Although my parents may not know it, this work would not have been started nor completed without their examples. To my father I owe my curiosity as to how systems function, and my desire to find structure in the world. To my mother I owe the ability to persevere in the face of seemingly insurmountable odds, and the core belief that I can succeed.

Finally, I would like to thank my companion in life, Johanna Moore, for her love and support - not least financial - throughout my graduate student career, her willingness to discuss my work when needed and her tolerance of the seemingly endless string of unrealistic finishing dates.

# Abstract

Recognition of motion sequences is a crucial ability for biological and robot vision systems. We present an architecture for the higher level processes involved in recognition of complex structured motion. The work is focused on modeling human recognition of Moving Light Displays. MLDs are image sequences that contain only motion information at a small number of locations. Despite the extreme paucity of information in these displays, humans can recognize MLDs generated from a variety of common human movements. This dissertation explores the high level representations and computational processes required for the recognition task. The structures and algorithms are articulated in the language of structured connectionist models. The implemented network can discriminate three human gaits from data generated by several actors.

Recognition of any motion involves indexing into stored models of movement. We present a representation for such models, called *scenarios*, based on coordinated sequences of discrete motion events. A method for indexing into this representation is described. We develop a parallel model of spatial and conceptual attention that is essential for disambiguating the spatially and temporally diffuse MLD data. The major computational problems addressed are (1) representation of time varying visual models (2) integration of visual stimuli over time (3) gestalt formation in and between spatially-localized feature maps and central movement representations (4) contextual feedback to lower levels and (5) the use of attention to focus processing on particular spatial locations and particular high level representations. Several novel connectionist mechanisms are developed and used in the implementation.

In particular, we present advances in connectionist representation of temporal sequences and in using high level knowledge to control an attentional mechanism. We show that recognition of gait can be achieved directly from motion features, without complex shape information, and that the motion information need not be finely quantized. We show how the "what" and "where" processes in vision can be tightly coupled in a synergistic fashion. These results indicate the value of the structured connectionist paradigm in modeling perceptual processes: no previous computational model has accounted for MLD recognition and we do not know how it would be approached in any other paradigm.

# Table of Contents

# List of Tables

# List of Figures

The day is committed to error and floundering; success and achievement are matters of long range.

*Goethe*

# 1 Introduction

The human visual system demonstrates a remarkable ability to discriminate between different types of movement. The classic illustration of this ability is Johansson's Moving Light Display (MLD) [Johansson, 1973; Johansson, 1976]. Reflective pads were placed at the joints of an actor dressed in black, and the actor illuminated. Films were taken of the actor walking, jumping and making various other movements against a black backdrop. When these films were shown to subjects, they all recognized the display to be of a person walking, jumping, *etc.*, but reported single frames to be meaningless patterns of dots[1]. A presentation time of no more than 200 msec of movement was sufficient for all subjects to correctly identify an MLD of a person walking. In forced choice experiments, all subjects accurately identified 6 human and 3 puppet generated gait patterns with a presentation time of 400 msec. Further experiments [Kozlowski and Cutting, 1977; Cutting and Kozlowski, 1977; Bassili, 1978; Poizner *et al.*, 1981; Tartter and Knowlton, 1981; Fox and McDaniel, 1982; Todd, 1983; Runeson and Frykholm, 1983; Sumi, 1984; Proffitt, 1988; Cutting *et al.*, 1988] demonstrated the generality, sensitivity and early development of this faculty.

The experimental set-up designed by Johansson provides an excellent vehicle for research in computational modeling of both low and high level biological vision. It incorporates elements of both static and dynamic vision, involves both shape and motion, requires integration of information over time as well as space, and uses as input visual frames that contain a minimum of information - essentially just the locations of ten or so dots. The last point is important for two reasons: 1) it reduces the low-level processing needed to a minimum, and 2) the displays are perceptually minimal, in that any significant change in the display, for example moving the dots to off-joint positions, results in degraded performance

[1] In the outside margin on each page of this document three MLD frames are shown, one from each of three human gaits. If the pages of the document are flipped at a rate of approximately 60 leaves per second, the three gaits should be distinguishable. When the speed is correct the apparent motion effect will cause a strong perception of motion as in a movie. To do this (1) put the document on a flat surface (2) using the left hand, grasp the right edge of the document between thumb (underneath) and forefinger (3) fold the grasped edge over towards the left edge (4) slowly slide the thumb leftwards across the edges of the pages, thus releasing them one by one (back to front) to fall flat, and simultaneously view the sequence of frames thereby revealed. To view the sequence on the even pages, reverse these instructions. The odd pages show samples from a male, the even pages show samples from a female.

[Cutting, 1981a].

The scientific motivation for modeling this kind of phenomenon is to understand better the computational processes necessary for general purpose vision. Modeling biological vision requires paying attention to: computational constraints imposed by the nature of the underlying hardware (neural circuits); perceptual data indicating the abilities of human vision; neurophysiological data indicating the types of neural circuits and architectures existing in the brain; and ecological considerations concerning the purpose of the organism and the environment in which it evolved. But our focus here is computation; Zen-like, we maintain awareness of the neurophysiological, perceptual and ecological environment, while concentrating our attention on computational problems and strategies.

## 1.1 Research Context

The work described in this dissertation is in part an elaboration of some aspects of Feldman's extension [Feldman, 1988] to the Four Frames model of the primate visual system [Feldman, 1985]. It is closely related to Olson's dissertation [Olson, 1989] which discusses an overall architecture of visual motion understanding, and elaborates the intermediate level. A fundamental aim in these works and in this dissertation is to generate sound computational structures and algorithms that do not violate any of the known scientific constraints.

There is no *a-priori* reason why biological constraints and motivations need be taken seriously in computer vision research. Why do so here? The broad answer is that biological systems are the only extant general purpose vision systems; and general purpose vision is a problem that computer scientists have been addressing for decades using structures and concepts motivated by the available hardware (digital computers) with partial success at best. A relevant question is whether the neural nature of the brain is of any importance computationally - perhaps it is just a particular hardware system that does not affect the computational strategies employed? We do not accept this view: it does not fit with the experience of biologists, which is that natural systems evolve to utilize the available resources in remarkably efficient and situation-dependent ways. In this case, the consequence is that it is highly likely that the computational architectures employed in the brain are designed around the computational properties of the basic processing and communication elements, respectively neurons and axons. Therefore, taking the biological evidence seriously is of great importance if research in computational vision is to help us understand how the human visual system works.

Furthermore, using a computational architecture modeled on brain circuits is a good idea both scientifically and technologically. There has already been fruitful scientific interaction between traditional computer vision and the newer connectionist motivations, for example the explosion of interest in Markov Random Field characterizations [Geman and Geman, 1984; Chou, 1988] and Hough transform networks [Ballard, 1984; Califano *et al.*, 1989; Cooper, 1989]. In the arena of technological innovation and development, there is increasing interest in and demand for real-time behavior in computational systems,

e.g., robots. We have an existence proof of the ability of neural networks to achieve general vision capabilities in real time; if we can understand the algorithms involved, we can build massively parallel machines on which to implement them [Bailey and Hammerstrom, 1986; Frazier, 1990].

## 1.2 Connectionist Models

The basic computational substrate adopted in this work is the connectionist model [Feldman and Ballard, 1982]. Connectionist models consist of simple computational elements (units) which communicate by sending their level of activation via labeled links to other elements. The units may have a small number of states, and compute simple functions of their inputs. Associated with each link is a weight, indicating the "significance" of activation arriving over that link. The behavior of the model is determined by the pattern of connections, the weights on the links and the unit functions. This is massively parallel processing (MPP).

The single most important contribution of this dissertation to the field of computer science is its furthering of our understanding of mechanisms and architectures for MPP. Massively parallel networks have interesting computational properties which are hard if not impossible to describe with languages designed for sequential computation. Our current terminology and language for describing MPP is barely above the equivalent of machine code for sequential processors. As new connectionist architectures are designed, analyzed and researched, new concepts and languages will be developed to describe them, and new computational structures and mechanisms with different properties from those known to date for sequential architectures will be articulated. This work contributes to that development. Learning algorithms for massively parallel networks has been the focus of much research [Hertz *et al.*, 1991]. Our interest is in designing representations and mechanisms for MPP, an approach known as Structured Connectionism [Feldman *et al.*, 1988]. Our claim is that Structured Connectionism provides an effective means for modeling the recognition of articulated motion, as embodied in Moving Light Displays.

The use of connectionist networks is entirely appropriate for constructing computational models of perceptual processes: a connectionist model may be considered an abstraction of the essential computational properties of the neural networks composing the brain. It is more likely that we will be able to successfully model a perceptual system if we use a computational model analogous to the one known to be used by that perceptual system. The objection may be raised that connectionist networks are not at an appropriate level for modeling perceptual systems, in the same way that describing a forest ecosystem in terms of biochemical interactions would be inappropriate. However, the reader should understand that the connectionist networks we use do not necessarily descend all the way to single neurons and axons. A connectionist unit in our view can be used to model a small neuronal circuit (where the result of the circuit's computation may be reduced to a single value), a single neuron, or indeed part of the dendritic tree. Furthermore, we are

not modeling high-level cognitive functions, such as planning or theorem-proving[2]. We are modeling perceptual acts which take on the order of one second to perform. Given that the effective switching time of neurons is on the order of 10 msec, allowing at most 100 sequential computations per neuron per second, it is clear that the brain must exploit massive parallelism to achieve perception.

## 1.3 The Structure of Visual Recognition

As with most attempts at computational modeling of perception, one of the hardest tasks is to delineate a suitable portion of the observed phenomenon to be modeled. It must simultaneously be sufficiently compact and well-defined as to admit to modeling given current computational equipment, current levels of scientific understanding of perception, and within a reasonable time-frame; and sufficiently broad as to be demonstrably of scientific merit. Choosing suitable boundaries requires an understanding of the task in which the modeled subsystem is embedded. The appropriate context for this discourse is the structure of visual recognition. We list the known major sub-tasks subsumed under the visual motion recognition task, roughly in order from low to high level:

- Low-level feature extraction: computing pre-attentive features in parallel across the visual field [Aloimonos and Shulman, 1989], including the achievement of appropriate invariances (for example, to translation, rotation and scale).
- Correspondence: in a sequence of images, establishing which features in a given frame map to which features in the subsequent frame [Ullman, 1979b; Olson, 1989].
- Segmentation: determining which features are a part of the same object or structure [Jain *et al.*, 1979; Koch *et al.*, 1986].
- Tracking: keeping the imaging system focused on the area of interest in the scene [Yarbus, 1967; Jenkin, 1986; Seibert and Waxman, 1989].
- Structure from motion: determining aspects of the underlying physical structure of an object from the motion of points on the object [Ullman, 1984; Prazdny, 1986].
- Intermediate-level feature composition: detecting spatially-composed features in the visual array of low-level features, including the satisfaction of simple spatial constraints between the parts comprising the feature [Hummel and Biederman, 1990; Zemel *et al.*, 1990].

[2]Inasmuch as high level inference activities depend upon pattern recognition, e.g., "I've been in this situation before" and knowledge retrieval, e.g., "This kind of situation has these kinds of consequences", it may well be the case that even high level cognitive modeling should take the neural character of the brain seriously [Derthick, 1988; Shastri and Ajjanagadde, 1989].

- Indexing: using the features to hypothesize high level objects and movements in the scene [Feldman, 1985; Lowe, 1985; Plaut, 1984].
- Viewpoint recovery: determining the 3D parameters of spatial relationship between the imaging system and the object [Plaut, 1984; Lowe, 1985].
- Attention: concentrating computational resources on particularly interesting areas of the scene or objects in the scene [Mozer, 1991; Rimey and Brown, 1990].
- Verification: confirming that the hypothesis generated by the indexing process is in fact present in the scene [Lowe, 1985; Huttnelocher and Ullman, 1987; Plaut, 1984].
- Planning: determining what order to attend to different locations in the scene in order to facilitate the indexing and/or verification tasks [Whitehead and Ballard, 1990; Rimey and Brown, 1990].
- Gestalt perception: forming the overall percept into a coherent whole, where each part of the scene is assigned to a high level concept [Koffka, 1935; Olson, 1989].

Although we can neatly list these problems as separate tasks, in fact they are tightly coupled and overlapping in a functioning visual system. For example, planning, attending and verification are obviously related; segmentation and feature extraction are aided by the contextual information provided by the results of the indexing and gestalt processes; correspondence is aided by feature extraction; tracking requires some pre-conscious attention mechanism at least. All of these processes are involved even in laboratory interpretation of Moving Light Displays. The task is to select a research problem that plausibly allows us to ignore some of these sub-problems and assume solutions for others, while leaving intact a real world phenomenon.

## 1.4 Scope

A research problem that fulfills these criteria is recognition of 400 msec Moving Light Displays generated from single objects moving parallel to the image plane. Of the sub-problems listed above, we shall deal with structure-from-motion, feature composition, indexing, attention, and gestalt formation (although the treatment of structure-from-motion is cursory). We ignore viewpoint recovery, verification and planning; and assume solutions to low-level feature extraction, correspondence, segmentation and tracking. The reasons for these choices follow. Essentially the strategy is to ground our problem at the low-level in established computational work, and bound it at the high-level by choosing a task that could not possibly involve the highest level processes.

The processes we have assumed solutions for have all been previously addressed within the framework of massively parallel processing. Correspondence has been dealt with

by Olson [Olson, 1989]. Segmentation has received much attention [Koch *et al.*, 1986; Koch *et al.*, 1989]. Tracking has been dealt with in [Seibert and Waxman, 1989]. Citing these works, we assume that the tracking system is keeping the imaging system fixated on the moving cloud of dots; the correspondence system is establishing an optic flow field for the moving dots, with the output in the form of trajectories very similar to those generated by Olson's networks; and segmentation is moot, since there is only one object in our MLDs.

Our reason for not treating verification and planning is that these processes require some hypothesis as to what is occurring in the scene: a verifier must have something to verify, and a planner some some interpretation of the scene with which to plan. In the laboratory setting, before presentation of the display, it is arguably the case that no particular hypothesis is active. It is exactly our purpose to explain on the processes that form that hypothesis.

We may ignore viewpoint recovery by dealing only with MLDs of objects moving parallel to the image plane. In this case, there is virtually no motion in depth. This simplifies the intermediate-level processing enormously, since it means that the distance between any two connected dots remains constant, even in the image (for example, between the wrist and elbow). Our solution is consistent with using *qualitative* depth information, i.e., information about ordinal depth, or at least a partial ordering of object depths, that can be gained from relative motion [Ballard and Ozcandarli, 1988], stereo disparity [Weinshall, 1988] or occlusion [Thompson and Whillock, 1988]; but in the main, we ignore problems involving 3D.

The distinction between indexing and verification requires some elaboration, since both are processes which match low-level data to stored visual models. The essential difference is that indexing is also a retrieval process. In its most general form indexing is the process of accessing a database of object/movement models and retrieving those models that account for the features in the image or sequence of images, ordered by likelihood. It uses qualitative features, and is parallel in the sense that the retrieval process computes the qualitative match between the features and all the object/movement models simultaneously. Verification is the process of matching a particular object or movement model to the observed data. It is a quantitative process and is sequential in the sense that only one[3] object or movement model may be matched to the data at any one time; because the computational resources devoted to the match are so extensive as to make duplication for parallel matching infeasible. The idea is that indexing quickly selects one or a few candidate objects/movements, which verification then confirms or, rarely, denies [Roberts, 1965; Plaut, 1984; Lowe, 1985].

## 1.5 Computational Issues Addressed

The problems that we have not treated are all low-level processes that have been satisfactorily addressed, to a first approximation; or high-level processes that plausibly are not

[3]Or at most a small number.

involved in interpretation of 400 msec Moving Light Displays. Substantial problems remain: structure from motion, feature composition, indexing, attention and gestalt formation, and at all levels, representation.

The major computational issues addressed in this dissertation are concerned with:

- Questions of sequence and time in the representation of multi-part coordinated movement. Indexing into these representations. Integration of information over time.
- Gestalt formation in hierarchical representations and between spatially indexed and non-indexed representations[4]. Contextual feedback from higher levels to sharpen lower level processing.
- Attention as a mechanism for mediating bottom-up and top-down processing between particular entities and spatial locations at different representational levels.
- Spatial and temporal integration of low-level shape and motion features to form higher level shape and motion features.
- Recovery of structure from motion - in this case, recovering the underlying stick figure from the motion of the dots.

In Chapter 3 we develop a number of solutions to the first three problems; instantiations of some of these solutions are used in the final implementation. Feature integration is addressed in Chapter 4, and a structure-from-motion proposal is outlined in Chapter 7.

The primary issue when modeling time-varying data is the representation of sequence and time. Interestingly, the very nature of our computational substrate (neurons and axons) suggests a representation: use the processing and communication delays in the substrate to represent the time constants in the models. This is appropriate for perceptual problems such as MLD recognition in which the time constants involved are on the order of tens of milliseconds, the same as the switching and communication time constants of the computational substrate. Given this basic representation of time and sequence, there remain problems of dealing with different tempos (e.g., walking slowly and walking quickly), organizing part-whole models of sequences, and using the models to provide priming. Integrating information over time is a related problem. Our architecture provides two mechanisms for temporal integrations, one applicable to a short time scale (tens of milliseconds), the other to a long time scale (hundreds of milliseconds).

Gestalt formation is perhaps the central problem in visual recognition. Given a set of active features, the interpretation problem requires that each feature be accounted for by some explanation, and that no feature support more than one competing explanation. For example, a wheel could be a car wheel or a bicycle wheel, but not both simultaneously. In essence, each feature must be bound to some high level explanation. The process

[4]Spatial indexing of a representation refers to its coding of location in the scene; this is to be distinguished from the indexing *process*, which seeks to identify what is in the scene.

of aggregating features into consistent groups is called gestalt formation, and occurs in all perceptual modalities. The problem is compounded in vision because the low-level features are spatially diffuse and their spatial location may be important, while the high level explanations must be central representations without any associated spatial location (or at most a very coarse one); so the same feature at different locations may need to be bound to different explanations. We develop a solution to the problem of gestalt formation between spatially indexed and non-indexed representations of time-varying phenomena. This solution turns out to be useful as the basis for a parallel spatial attention map.

Perceptual attention is the name given to a complex of processing mechanisms whose fundamental aim is to increase the power of a resource-limited perceptual system by selectively assigning computational resources to particular tasks. We give the following general definition:

> Perceptual attention is defined as a set of processes or mechanisms which, given many lower-level representations, mediates access to and response from resource-limited higher level representations.

One such mechanism in the human visual system is head/eye movements. Here the limited resource is foveal vision, and the lower-level entities are the different spatial locations in the scene at which objects may be found. These external attention shifts take hundreds of milliseconds to execute (typical saccades take 150 msec just to initiate [Lisberger *et al.*, 1987]). Since simple MLDS can be discriminated with as little as 400 msec of presentation time [Johansson, 1973], we may safely assume that the contribution of external attention shifts to processing is minimal in this case[5].

Another type of attentional mechanism in the human visual system is the internal attention shift [Shiffrin, 1988], i.e., one executed in the network circuitry. This has generally been conceptualized as an analogue of the external attention mechanism, resulting in the "spotlight of attention". This mechanism appears to take anywhere from 50 to 500 msec to execute an attention shift, but the data indicating the lower (50 msec) figure are from visual search experiments, and may be more indicative of the extent of parallel processing of different objects rather than timing of internal attention shifts. However, it is possible that this mechanism could play a role in the indexing, although it is unclear what that role would be. The attentional mechanism we propose could implement these internal attention shifts if an appropriate control strategy were developed.

There are two key characteristics of these mechanisms that are often assumed to be essential characteristics of visual attention 1) sequentiality - only one entity at a time is selected, and 2) spatiality - the selected entity is a spatial location. For example, a saccade selects a single spatial location in the scene for foveal vision to operate on. However, it is possible to envisage attention mechanisms that can select multiple entities in parallel,

[5]For example, there would be time for a single saccade to the centroid of the cloud of moving dots.

and that are not restricted to selection of spatial locations alone[6]. We explore one such mechanism that includes sequential attention to single spatial locations as a special case.

The visual indexing problem imposes a tension between the desire for complex features which are less ambiguous relative to competing interpretations and the desire to minimize the number of features computed at each of the many spatial locations. We develop a multi-resolution hierarchy of representations, the lowest levels of which contain features unrelated to any particular object, and the highest levels of which contain features specific to at most a few objects. The hierarchy has the novel aspect of being multi-resolution in time as well as space, giving an implicit mechanism for integrating information over time. It is organized in two pathways, one for shape information and the other for motion information, consistent with the architecture of the primate visual system [Maunsell and Newsome, 1987]. The shape and motion features are integrated at a high level in a third hierarchy that represents named movements (e.g., "running"). Translation invariance is achieved by duplication of feature detectors across the visual field. We assume that lower-level processes have normalized visual size, so that scale invariance is moot. Rotation invariance is not required, since the data indicate that humans have great difficulty in recognizing inverted MLDs [Sumi, 1984].

The lowest level problem that we address, recovering the structure of the underlying stick figure from the motion of the dots, is one that others have worked on [Rashid, 1980; Webb and Aggarwal, 1982; Hoffman and Flinchbaugh, 1982]. These solutions are sufficient for the task at hand, but comes nowhere close to accounting for human abilities (see e.g., [Braunstein, 1976; Doner *et al.*, 1984; Todd, 1985]) and inabilities (see e.g., [Braunstein, 1986; Dosher *et al.*, 1986; Todd *et al.*, 1988]). The problem of constructing a general model of human recovery of structure from motion is well beyond current understanding. In the next Chapter we treat the structure-from-motion problem in the context of the overall architecture of our model. Further discussion of this problem is left to the section on future work in the final chapter.

## 1.6 What and Where formulation of the problem

One of the classical ways to divide up the vision problem is to separate out the "what" and "where" tasks [Mishkin, 1982; Mishkin *et al.*, 1983; Mishkin and Appenzeller, 1987; Maunsell and Newsome, 1987]. The "what" task is to determine what is in the scene. The "where" task is to determine where the different objects are located in the scene. If the answer to one of these questions is known, the answer to the other is much more easily computed. In a scene with one recognizable object and some clutter, if the objects's location is known then its identity is much easier to determine (by zeroing out other locations). If the identity of the object is known, it is much easier to find its location (by matching the object against the image patch at each location). If neither is known, then potentially each

[6] Although selection of spatial location seems to be a necessary part of visual attention.

modeled object must be matched against each spatial location. One of the novel aspects of the work reported here is that as the models of visual sequences are indexed (answering the "what" question), a representation of where each sequence is occurring is constructed in the spatial attention map (answering the "where" question). The processes that solve these problems act with synergy. It is not the case that one of the problems is solved before the other, but rather that they are solved simultaneously with partial solutions to the one enhancing the solution to the other. Thus the "what" and "where" processes are brought together in an attentional mechanism [Kosslyn, 1987].

## 1.7 A Summary of the Dissertation

In this introduction we have outlined the perceptual phenomenon we are attempting to model, our motivations for doing so, and the major problems involved. We have listed those problems for which we assume solutions, and given an overview of those we address.

In the next Chapter, we describe the overall architecture of the model, and discuss related work. In Chapter 3 we develop connectionist mechanisms for representing visual sequences and activating those representations and for forming gestalts between representations with varying degrees of spatial indexing or none at all, and for attending to particular spatial locations and sequences. In Chapter 4 we describe the network implementation of the representations of gait models, the multi-resolution feature hierarchy, and the interactions between the central gait representation and spatially indexed feature representations. Chapter 5 describes pilot experiments and results using small sets of hand analyzed gait data. Chapter 6 describes the main experiments using a large amount of gait data obtained with an automated gait analysis system. The summary, conclusions and directions for future work appear in the final Chapter.

# 2 Architecture

This chapter describes an architecture for a part of the visual system which is involved in recognition of moving light displays. It is one of many possible architectures; however, it conforms to the constraints that any architecture must satisfy if it is to be capable of solving the MLD recognition problem. The input to the model is the output of a hypothesized low-level system which computes an augmented optic flow field very similar to the segment/trajectory level in [Olson, 1989]. The output is a decision as to which movements are occurring in the scene, *including an encoding of where the movements are* taking place.

The architecture is shown in Figure 2.1. On the left are the names of the different levels of the architecture, and a pictorial example of the the kind of entity found at each level. There are three parallel part-whole hierarchies: Shape, Motion and Scenario. Shape and Motion represent uninterpreted features in the scene, for example a pair of lines of particular lengths whose ends join at a particular range of angles. Scenarios represent named sequences of Shape and Motion features that constitute a gait or other complex movement, for example "biped-walking". Generally scenarios are higher-level entities than features, but there may be some overlap as illustrated in Figure 2.1. The augmented optic flow input is transformed into low-level shape and motion features, which propagate up their respective pathways. At a relatively high level, the shape and motion features begin to activate the lowest level of the scenario hierarchy, up which the activation propagates. The network which implements the architecture is assumed to have an update rate of a few hundred steps per second.

In this chapter we describe the kinds of entities represented at different levels of the hierarchies in each pathway, and justify the choice of levels and modalities of representation. We discuss the constraints which led us to this type of architecture, and the trade-offs that must be addressed. Finally, we review previous work which is related to this model.

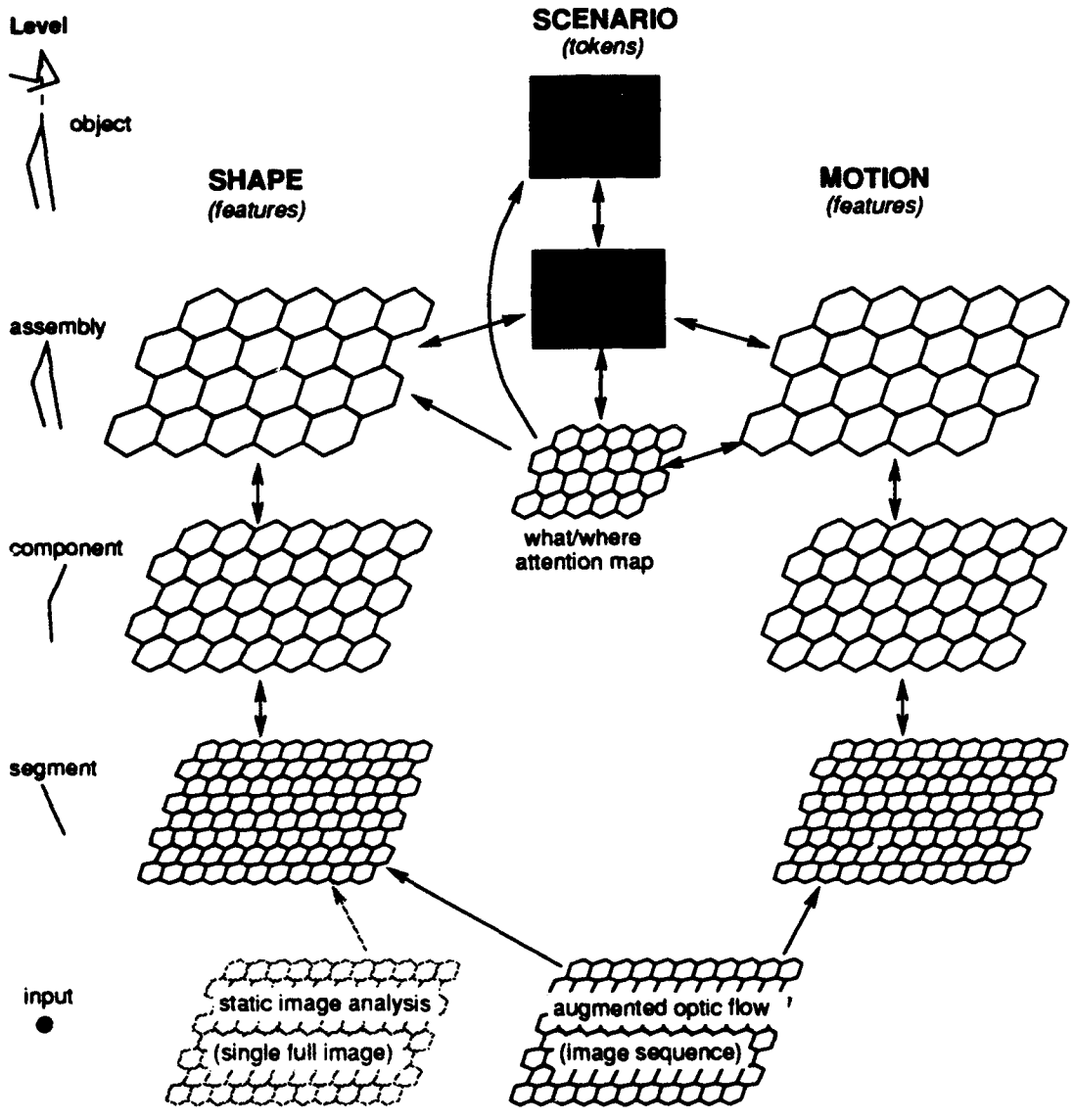


Figure 2.1: Architecture Overview

## 2.1 Structure From Motion

The input to the model is in the form of augmented optic flow. The stimuli are moving light displays, which consist of a dozen or so moving dots. The input information consists of, for each dot, its location, speed, and the approximate radius and direction of curvature of the current path of the dot; essentially, Olson's *trajectories* [Olson, 1989]. The first step is to recover the structure (connectivity) of the MLD. The structure is recovered in the form of line segments (joining two dots), both in terms of shape (length, orientation) and motion (rotational velocity). The line segment features form the lowest levels of

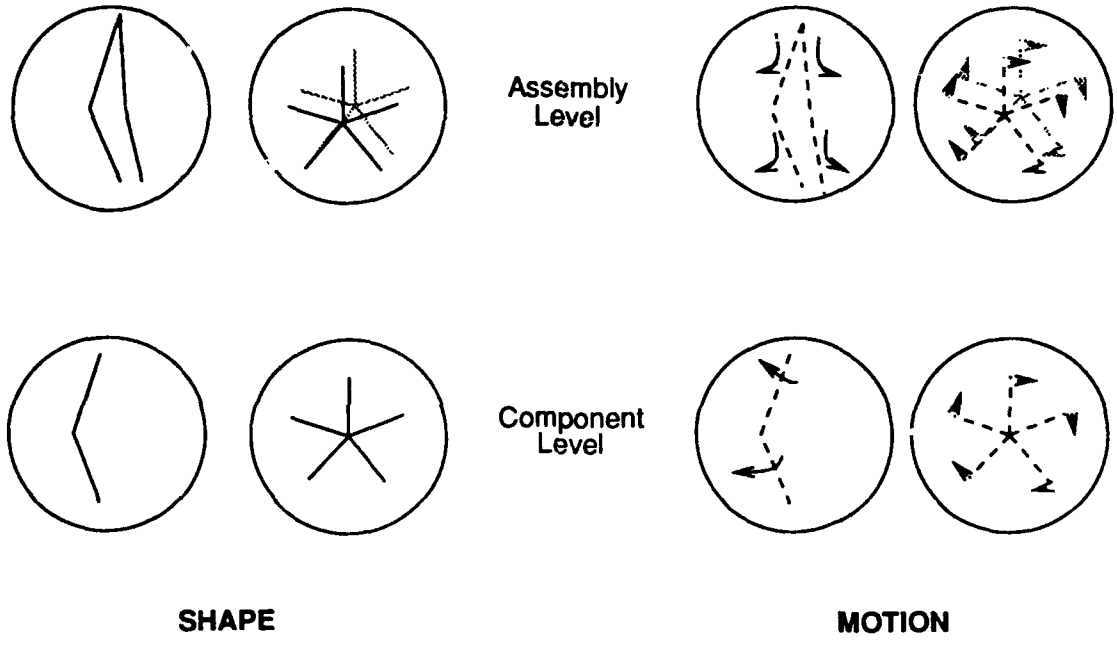


Figure 2.2: Features in the Shape and Motion Hierarchies

the shape and motion pathways. We make no claims as to the correctness of this part of the architecture as a cognitive model. It represents a subset of the information that must be computed in the visual system, but is too simple to account for any significant portion of human ability to recover structure from motion [Wallach and O'Connell, 1953; Prazdny, 1986]. Further discussion of the structure-from-motion problem is left to the final chapter.

A few words about frames of reference are in order at this point. Olson asserts that his trajectory level (our input) is computed in a viewer-centered coordinate system (the Stable Feature Frame [Feldman, 1985]). The shape and motion hierarchies introduced here are computed in the same reference frame. We assume that the imaging system is tracking the MLD. We further assume that during tracking the Stable Feature Frame is identical with the retinal frame. The effect is as if the MLD had its translatory motion zeroed out ("walking on the spot"), and then all processing is done in retinotopic coordinates.

## 2.2 Shape and Motion Pathways

The Shape and Motion pathways compute and represent general features of the scene. As we move up each hierarchy, the features become spatially more extended and compositionally more complex. At higher levels the features correspond quite closely to particular objects or gaits, e.g., *quadruped rear legs*. For temporally continuous motion, the higher the level, the greater the temporal interval over which a feature responds - so that features exhibit

increasing temporal extension as we move up a hierarchy. Each Shape or Motion feature at a particular level represents a range of shapes or motions that an object at that level may express. Figure 2.2 shows examples of the kinds of shapes and motions found at the two levels used in this system. This figure shows features derived for legs and for spoked wheels, although only the former were implemented[1]. The fact that there are only two levels of features above the simple line segment level is an arbitrary choice made for the implementation. In biological systems, the distinction between levels may not be quite so clear, and there may be more levels.

The Shape and Motion pathways are spatially indexed, as indicated by the hexagonal tiling in Figure 2.1. They are spatially indexed so that a level *n* feature may verify the spatial relations between the level *n-1* features of which it is composed. This implies that there is spatial coherence in the Motion pathway, a prerequisite if any sorts of complex motion features are to be computed and represented.

Activation provided by low-level features propagates up the hierarchies, causing higher level features to become active. As this happens, the high level features compete to "own" the activation from the low-level. Simultaneously, the higher levels provide facilitation to those low-level features with which they are consistent. Chapter 3 describes mechanisms to achieve this kind of feature-binding [Olson, 1989].

## 2.3 The Scenario Hierarchy

The purpose of Scenarios is to represent temporal series of events. They represent information about *sequence* and *duration*. A Scenario consists of a set of labeled events, a temporal ordering on those events, and a specification of the interval between consecutive events. While activation of a Shape or Motion feature requires nothing more than that the appropriate elements be present in the scene *at the current time*, activation of a Scenario depends upon the *history* of Shape and Motion features up to this point in time. Scenarios represent interpreted histories of change (in this case, visual change) and are relatively complex. It would seem extravagant to reproduce information about sequences and durations in many locations; therefore the representation for scenarios is not spatially indexed. This lack of spatial indexing results in cross-talk problems if there is more than one Scenario occurring in the scene which, as we shall see, is the case for bipedal gait. In Chapter 3 we explore mechanism for ameliorating these cross-talk problems.

The lowest level scenario in Figure 2.1 overlaps the highest level of the shape and motion pathways. At this level, the scenarios represent the temporally-extended movements that objects at that level may make. At higher levels, the scenarios represent temporally and spatially coordinated combinations of lower-level scenarios. Figure 2.3 illustrates the kind

[1] Although wheels have featured little in the psychological literature concerning MLD recognition [Cutting and Proffitt, 1982], they are easily recognized. Many of the obvious representations for animal motion totally fail for MLDs including wheels with dots at the center and on the circumference.

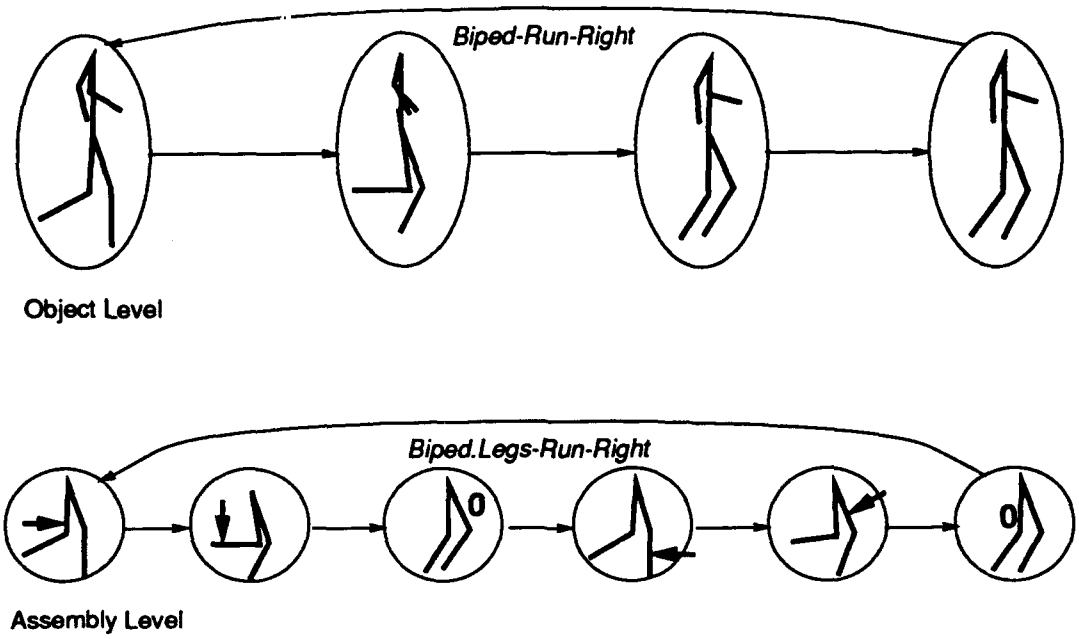


Figure 2.3: Scenarios at different levels

of scenarios that occur at each of the two levels shown in Figure 2.1. Assembly-level events detect specific changes in motion, as shown in the Figure. For example, the first event in the *Biped.Legs-Run-Right* scenario is detected when the rear thigh begins to move counter-clockwise.

These figures are illustrative of the system built as part of this thesis research, to recognize moving light displays. The particular levels at which scenarios commence and the level at which features terminate depend upon the individual system. For example, a choreographer could be expected to have detailed scenario representations down to a low level for human body movements and up to a high level for coordinated human dances, but be unable to distinguish (name) different types of animal gait; whereas a jockey might have more detailed representations of various types of horse movement at several levels, but be unable to distinguish different dance steps or dance sequences.

## 2.4 What/Where Attention Map

The what/where attention map (WWAM) is a representation of which scenario is occurring where in the scene, and is the basis for an attentional mechanism that can focus bottom-up and top-down processing on particular spatial locations and on particular scenarios. The map is based on a simple idea: measure the correlation over time between the activity in each scenario and the activity of the features at each location. If a scenario has low activity - it has not been selected by the indexing process - then the correlation is low everywhere.

If a scenario is becoming active - the changes in activity of the features at some location correspond to events represented in the scenario - then the correlation at that location begins to rise.

As a correlation between scenario events and feature activity at a given location increases, this fact is used to strengthen the contribution of the features at that location to the scenario, and to increase the contextual feedback that the scenario provides to those features. The correlations are computed in parallel, requiring resources of order (number-of-scenarios x number-of-locations). This is a reasonable demand since the spatial resolution at this level is quite coarse. In the implemented system we use the same resolution in the attention map as in the feature assembly, but the attention map could easily be modified to utilize a coarser resolution.

## 2.5 Related Work

The central questions addressed in this dissertation concern: recovery of structure from motion, formation of complex spatial features, representation and indexing of movement models, gestalt formation in and between spatial and non-spatial hierarchies, and attention to focus processing on particular locations and scenarios. In this section we discuss previous work related to these problems and our architecture.

### 2.5.1 Structure from Motion for MLDs

The classic paper on structure-from-motion is [Ullman, 1979a]. In that paper it is proved that:

> *Given three distinct orthographic views of four non-coplanar points in a rigid configuration, the structure and motion compatible with the three views are uniquely determined.*

In further work [Ullman, 1984], an incremental scheme for recover of rigid structure from motion is proposed. It relies on an internal interpretation which is updated over time. If new information arrives that is inconsistent with the current model, then the model is changed in a minimal way to fit the new data. A variant using a flexible models with springs seemed to perform better in experiments. This incremental rigidity model was psychophysically evaluated [Hildreth *et al.*, 1989] and found to correspond qualitatively with data from human subjects, which showed incremental buildup of structure, plateauing after about 1 second. The main difference was that in the HVS the initial interpretation is better than that in Ullman's model, and the final interpretation in the HVS worse than that in the model. In any event, we do not have rigid motion in MLDs (except between certain pairs of dots), so these results are not directly applicable. However, there has been structure-from-motion work specifically directed at MLDs.

In his dissertation, [Rashid, 1980] presents an algorithm which segments the points of an MLD into body parts. His system, *Lights*, solves the problems of dot correspondence between successive frames, grouping of dots into distinct objects, and finding the skeletal structure (connectivity) for each object. The computation of connectivity is relevant to our desired transformation of dot velocities into relative motion parameters. If connectivity can be established, it is easy to derive the relative motion parameters. Rashid performs a graph-theoretic cluster analysis on the dots' position and velocity parameters in order to find the probable connectivity. His distance function is a modified Euclidean metric accumulated over successive frames. The algorithm has problems because the arm and leg movements on opposite sides of the body can be very similar, so that they are clustered together.

[Hoffman and Flinchbaugh, 1982] set out to compute the three dimensional structure and motion of the figure giving rise to the MLD. Somewhat as a side effect, the skeletal connectivity of the figure is computed. They prove two "structure from planar motion" propositions, assuming that the motion of the figure is in a plane (not necessarily the image plane). The two propositions are:

1. *Given three distinct orthographic projections of the two endpoints of a rigid rod which is constrained to move in a plane, the structure and motion compatible with the three views are uniquely determined (up to a reflection about the image plane).*

2. *Given two distinct orthographic projections of the three endpoints of two rigid rods linked to form a pairwise-rigid structure which is constrained to move in a plane, the structure and motion compatible with the two views are uniquely determined (up to a reflection about the image plane).*

A pairwise-rigid structure is a set of points moving in space so that each point remains at a constant distance from at least one other point, and no three points are in a rigid configuration. Interestingly, the proof of these propositions requires that the rod or pairwise-rigid structure of two rods be moving in some plane *other than* the image plane. If the motion is in the image plane, then the there are two possible solutions to the structure and motion equations (up to a reflection about the image plane). The planarity scheme has been implemented in a small network of complex nodes. The idea is to submit successive pairs and triples of MLD dots to the implementations of the propositions, to find out which are linked. This *sequential* processing is in contrast to our approach.

Both of the approaches to the recovery of MLD structure described above are completely bottom-up. No models of object motion are presented, and consequently there is no top-down contextual information to aid the computation of structure. [Hoffman and Flinchbaugh, 1982] discusses this:

> *In general, bottom-up avenues of interpretation should be exhausted before recourse to top-down schemes is taken. With this consideration in mind, a second interesting possibility exists. Perhaps the limb structure and motion*

*obtained bottom-up using the planarity assumption is sufficient to provide a unique index into a stored table of 3-D models of animals. The interpretation of biological motion would then involve an interaction of both bottom-up and top-down processes. The bottom-up processes get the interpretation process off the ground and the top-down processes complete the interpretation of those structures which resist bottom-up attack.*

It is precisely this "second interesting possibility" that we have explored. Our architecture constructs a high-level interpretation of movement that is then used top-down to refine the lower-level processes. Although the top-down flow of contextual information does not extend to the structure from motion process in the implementation, there is no reason why it could not do so. This is one of the most promising avenues of future work explored in Chapter 7.

[Webb and Aggarwal, 1982] presents an algorithm for computing the 3-D structure of a pairwise-rigid structure that uses the assumption that rotation of each rigid part is about a fixed axis, which is a generalization of the planar-motion assumption used by Hoffman and Flinchbaugh. Webb and Aggarwal also suggest that it "seems likely that some high-level knowledge is used by humans in interpreting [MLD] images".

### 2.5.2 Multi-Resolution Hierarchies

Multi-resolution hierarchies, are increasingly used in image processing [Uhr, 1987; Tanimoto, 1989]. Various kinds of processing and representational structures are labeled "multi-resolution", but the underlying idea in all cases is to analyze the image at several different resolution levels, with coarser resolutions used for larger scale features. The essential advantage of this kind of representation is that it provides a mechanism to make the trade-off between the competing demands of fine spatial resolution and low computational complexity. Visual features occur on vastly different spatial scales, even within the same image or object. If a single spatial resolution is used, then it must be as fine as that required by the smallest feature, but then even the coarsest feature is computed at that fine resolution. It is more appropriate, and efficient in the amount of computation required, to compute small scale features (e.g., a line segment) at a fine resolution and large scale features at a much coarser resolution.

Generally the multi-resolution approach is used to focus processing at the appropriate level, an attention-like mechanism. This introduces an element of sequentiality into the process of feature extraction. For example, in [Dyer, 1987], a coarse-to-fine search strategy is employed, using information computed at a coarse resolution to control the computation at a fine resolution. In our terminology, Dyer's system is performing the initial *indexing* with coarse features, and then *verification* with the coarse-to-fine search process.

Some multi-resolution processing structures are also known as pyramids [Tanimoto and Klinger, 1980]. In [Tan and Martin, 1986], a *pipelined pyramid* is introduced to deal

with dynamic imagery. At successive time intervals, the next frame is inserted into the base of the pyramid (the finest resolution level), and concurrently each succeeding level computes its features from the immediately preceding level. Thus for an *n*-level pyramid, there is information spanning *n-1*-intervals in the pyramid at any given instant. [Tan and Martin, 1986] also introduce attentional processes that have the ability to examine arbitrary sub-images at arbitrary levels of the pyramid, which use a coarse-to-fine strategy to verify hypotheses generated from features at the coarsest resolution level. Our architecture is a form of pipelined pyramid, but we have a different kind of attentional process.

Connectionist models that have used multi-resolution hierarchies have not introduced this sequential aspect to the processing. In [Honavar and Uhr, 1989] the network learns to construct more complex features at each succeeding level. In [Hummel *et al.*, 1989] the network is pre-wired to be sensitive to the particular types of features found in line drawings of objects. In both cases, feature computation proceeds in parallel at every location and at every level. Our architecture is related to that found in [Hummel *et al.*, 1989], in that we pre-wire the feature detectors, and use overlapping hexagonal receptive fields. However, our features at the component level and above are not general features that *might* occur in the world, but features that correspond to one or more objects *known* to occur in the world. This distinction is important, because there are many more possible arrangements of things than actually occur in the world. For example, although leg like motion-features could occur in many configurations, in fact they only occur in a small number of configurations. *Our feature hierarchy represents only these legal configurations.*

### 2.5.3 Visual Recognition

The seminal work [Roberts, 1965] remains the foundation for computer vision research today. The basic ideas of extracting local features, combining them to form symbolic descriptions, and matching these descriptions against a database of models have been enriched by subsequent work. The *reconstructionists* [Marr, 1982] have attempted to develop mechanisms to recover scene features and structure in a bottom up fashion. Eventually the ill-posed nature of the problem thus formulated led to *regularization theory* [Poggio *et al.*, 1985], in which smoothness or other assumptions about the nature of the scene are used to transform the problem into a well-posed one, and to *fusion* approaches [Aloimonos and Shulman, 1989] in which the scene is analyzed in multiple modalities (e.g., shape, color, motion, texture) each of which is under-constrained but which can be combined to provide a single interpretation.

A much more compelling approach based on consideration of environmental factors was put forward in [Gibson, 1950]. This approach considers the vision problem in the context of the goals of the organism, the nature of its environment, and the organism's ability to intentionally structure its interaction (visual and otherwise) with that environment. The recent upsurge of interest in *active vision* [Ballard, 1990] is a consequence of not only technological advance (e.g., digital stepping motors, small CCD cameras), but also the

realization that problems which are under-constrained in static scene analysis can be easily solved using motion, tracking or other goal-directed behavior (e.g., [Ballard and Ozcandarli, 1988]).

The direct antecedents for this thesis are the Four Frames article [Feldman, 1985] and its extension to deal with time [Feldman, 1988]. An early development of static object recognition within the Four Frames paradigm is reported in [Plaut, 1984]. In a "tiny world", he implemented the indexing and verification phases of the recognition process. [Cooper, 1989] also reports object recognition work within the Four Frames paradigm. Cooper's dissertation presents, amongst other mechanisms, a connectionist object recognition network, working in the Tinker Toy domain. The same basic ideas of parallel matching against 2-D prototypical views of stick figures is used here.

The analysis of visual recognition underlying Cooper's work and this thesis is dealt with extensively in [Lowe, 1985]. Perceptual organization of visual information generates features which are used to index into 2-D models. These 2-D models are linked to 3-D models. Once the most likely candidate has been identified, the idea is to determine the viewpoint and perform a verification phase in 3-D. The advantage of this approach is that while reconstruction of 3-D information for the initial indexing phase is not always possible and even when possible often not very reliable [Lowe, 1985], the 2-D features are easily computed and often sufficient to reliably suggest the correct candidate 2-D model. Since biological vision systems are not equipped with laser range finders or other precise depth sensors, the most reliable and immediate information available is that on the retina: 2-D projections of the scene.

### 2.5.4 Time-varying Imagery

The earliest work on high-level interpretation of time-varying imagery that is relevant is Badler's dissertation [Badler, 1975]. A framework is described in which short movie sequences (which he calls scenarios) can be described in linguistic terms. Each frame is analyzed to find the objects in the scene, then "events" are detected whenever there is any significant change: a new object appears; an object starts to move; the agent or instrument of motion changes; there is a change in direction or velocity of motion, or in axis of rotation; and others. Events can be arranged in sequences and a sequence subsumed under a higher level event. At the top of the event hierarchy are nodes that correspond to the motion verbs. Event duration in the scene is represented by the number of frames an event endures, but there is no check for compatible time intervals when events in the scene are matched against stored models. Sequence matching is performed by considering each previous event as a possible start point for the sequence, which is computationally expensive.

Despite the apparent similarity between Badler's work and our model, his framework is not sufficient for our task. The major differences are in the degree of static scene analysis before temporal integration begins, and in the attention paid to issues of time. Badler performs extensive analysis on each frame, to the point of identifying the objects

in the frame, before attempting to integrate the current frame's analysis with those of previous frames. In contrast, our features are not object specific and are not as complex as Badler's objects. His sequence matching essentially ignores the problems of representing and matching time intervals, whereas in our scenarios there is extensive machinery for dealing with time. Other problems are that in Badler's framework no interpretation is given until the end (or one cycle) of a modeled sequence, and no top-down predictive capability is possible until one cycle is complete. Recall that MLDs are recognized when about one half of the gait cycle is presented. Finally, Badler's framework was not implemented.

A related piece of work implements part of Badler's framework as the analysis component of a perceptual system [O'Rourke and Badler, 1980]. This system uses a detailed model of human body motion to analyze digitized images sequences of human motion. It consists of four stages arranged in a cycle 1) **analysis** - given a set of predicted feature locations, find the actual locations in the new frame 2) **parse** - chunk the feature motion into linear segments (constant velocity or angular velocity) 3) **predict** - produce a prediction of future motion 4) **simulate** - given the prediction from stage 3, project back to image coordinates to provide a new set of predicted feature locations for the next iteration of step 1 to use in the next frame. The output is a track of the individual body parts plus a sequence of primitive motion chunks. Stage 3 is where a detailed model of human gait could be used, but the work says nothing about the indexing problem - how to determine the object and the gait - which is the major focus of our work. However, O'Rourke's system has two computational features similar to ones used in our model: multi-resolution analysis in space and, simply, in time; and a part-whole hierarchy in space (e.g., the body model includes the arm model) and in time (sequences of motion can be composed). The major point of interest in these two features is the idea that some of the benefits of multi-resolution analysis and part-whole hierarchies that are wel' known in the spatial domain have their counterparts in the temporal domain.

The most ambitious work to have grown out of Badler's framework, and perhaps the closest work to our own, is that by Tsotsos [Tsotsos *et al.*, 1980; Tsotsos, 1987]. The later work presents a knowledge-intensive framework that integrates time into high-level (attentive) vision. It incorporates a parallel network of hypotheses arranged in cooperative and competitive subnets. An iterative control mechanism propagates certainties locally amongst hypotheses, accepts new data from the image analysis section (feature extraction), and repeats. Dynamic weights that depend on the current set of active hypotheses influence certainty propagation. Certainties accumulate in a hypothesis, under complex rules, until an "activation" level is reached, at which point the hypothesis becomes active. Temporal sequence representation is based on events which have a given duration. Priming is passed from one event to the next, decaying exponentially with time. Overall, it would appear that the framework could be reformulated as a structured connectionist network with similarities to our own model, without great difficulty. However, the treatment of time is deficient - priming from one event to the next should depend on a time interval, which is poorly modeled by exponential decay. Our scheme for representing and activating sequence models (scenarios) is much richer and more flexible. In addition, the question of spatial

indexing is treated quite differently in the two frameworks. Tsotsos's model is mainly post-attentive, meaning that the spatial location of event hypotheses are data-driven. In contrast, our model is mainly pre-attentive, with spatially indexed feature maps indexing central scenario representations in a parallel fashion, and even our attention mechanism operates in parallel. Tsotsos's model could accommodate the aspects of lower-level vision that we have tried to capture with our feature maps, but it would require hard-wired nodes for events at each different spatial location rather than creation on the fly[2].

A system for tracking a person walking was developed and described in [Hogg, 1984]. The person is modeled as a part hierarchy of cylindrical elements, each with its own coordinate system. Walking is modeled as a set of cubic B-splines of one periodic parameter (phase), one spline for each of the four joints (shoulder, elbow, hip, knee). The models are used to predict the next image, and then the prediction is matched against the next frame. There is no attention paid to the indexing problem, and no obvious way in which the model of walking could be of use in the indexing problem. Another tracking algorithm [Jenkin, 1986] was developed for MLDs, but it provides no clues as to how the indexing problem could be solved.

A general framework for representation and recognition of the movement of shapes was presented in [Marr and Vaina, 1982]. The suggested representation for motion sequences as state-motion-state is close to the event-interval-event representation we have adopted for our scenario representation.

Recently there has been some attention paid to the idea of using motion itself as the basis for recognition of structured movement [Goddard, 1989; Gould and Shah, 1989; Allmen and Dyer, 1990]. This is in contrast to the approach taken previously, which has been to perform static analysis on each frame (sometimes using motion as a cue, e.g., for segmentation), and to integrate the frames at a high level. In [Goddard, 1989], we described an early version of the model presented in this dissertation, suggesting that motion features be used directly for recognition of movement. The *Trajectory Primal Sketch* [Gould and Shah, 1989] is an attempt to extend the notion of a primal sketch [Marr, 1982] to include motion features. These features are based on significant changes in motion, i.e., rapid change in velocity, and can be invariant to transformations such as rotation, scale and translation. A similar idea underlies our choice of events in the scenario representation. ST-curves and ST-surfaces [Allmen and Dyer, 1990] are representations of complex motions which are proposed as the basis for recognition of structured movements. An invariant form, curvature scale-space, is developed, which is used to match movements such as wing-flapping. However, many frames must be recorded to extract the ST-curves, so that the scheme is not incremental. Moreover, a complete cycle of motion has to be seen before matching can be done.

[2]Personal communication.

### 2.2.5 Attention

In the psychology literature, there is a large body of work on both eye-movements and internal (covert) attention mechanisms [Shiffrin, 1988]. There has not been commensurate interest in the computer vision community, until recently. The "active vision" paradigm has emphasized sensor control (e.g., eye-movements) as a means of enhancing the power of a perceptual system, but has largely ignored internal attention mechanisms. A review of this literature may be found in [Rimey and Brown, 1990]. We have argued that, at least for recognition of 400 msec MLDs, sequential eye-movements cannot play any significant role since they take several hundred msec to execute.

Covert attention mechanisms have been proposed in [Koch and Ullman, 1985; Mozer, 1988; Ahmad and Omohundro, 1991; Tsotsos, 1991]. In [Koch and Ullman, 1985], an attentional spotlight of fixed size is implemented with a simple winner-take-all network. The attentional mechanism uses a topographic map of units to represent the focus of attention. These units gate the flow of activation from low to high level representations. There is no top-down control based on semantic information; it is a data-driven system. MORSEL [Mozer, 1988; Mozer, 1991] is a connectionist object recognition system that incorporates a variable size attentional spotlight which is implemented with a topographic map of gating units. There is provision for top-down influence on the region of attention, but no indication of how the top-down information is derived in a visual recognition task. SWIFT [Ahmad and Omohundro, 1991] is a search strategy implemented in a connectionist model of visual search. Once again, the attentional mechanism is a topographic map of gating units that implements a variable size focus. In this system, however, control of attention is both data- and semantically-driven: a priority network combines low-level feature information with high-level task information to arrive at the next focus location. Tsotsos' proposed attentional beam [Tsotsos, 1991] is a complex iterative mechanism for localizing stimuli using a variable sized spotlight of attention. The size, shape and location of the spotlight is refined by the context developed in the previous iteration. The mechanism uses computing units that produce several independent output values, violating a basic connectionist assumption (one unit, one output), and is not implemented. In all of these models, the object of the attentional system is location selection, and the mechanism is a topographic map gating units. In our model, we too use a map of gating units, but the top-down feedback allows both location and high-level representations to be selected, thus binding particular high-level entities to particular locations; and this occurs in parallel. In addition, our model uses partial results from the recognition process to drive the attentional mechanism, which has not been done before in connectionist models.

### 2.5.6 Connectionist Models of Sequence and Time

There is an increasing amount of work on temporal sequences in connectionist network research. Overviews may be found in [Pearlmutter, 1990] and [Hertz *et al.*, 1991]. Here we review the main ideas. The most relevant connectionist work on sequences has been

in speech perception [Lippmann, 1989], a domain in many ways analogous to ours. The similarity is in the necessity to treat time as a critical factor. The major difference is that in the speech perception there is a single time-varying signal; in the visual domain, we have multiple interacting sequences.

The representation of time in connectionist networks has been approached in two ways: time converted to space, and time represented as processing delays. The first approach allocates different input units to different time slices (a buffer), effectively converting a one-dimensional temporal signal into a two-dimensional spatial pattern [Cottrell, 1985; Hanson and Kegl, 1987]. This approach has several drawbacks [Mozer, 1989]: the buffer size is fixed in advance; many input units are needed; crucially, the time at which to start filling the buffer is hard to determine, especially before the signal is available (the segmentation problem). The second approach represents time implicitly as communication delays along the links connecting the connectionist units, and processing delays within the units[3], and keeps at most a very limited history of prior inputs. Learning time delays has been treated in [Bodenhausen and Waibel, 1991]. We have adopted the time-delay approach, and restrict our review to this approach.

### Attractor Networks

[Kleinfeld, 1986] and [Sompolinksy and Kanter, 1986] have developed sequence storage and retrieval models using spin-glass type networks. This type of network, after initialization to some state, evolves through stable attractor states at each iteration, the states forming the sequence. The temporal qualities are achieved through links with associated delays. These models are extensions of the Hopfield type [Hopfield, 1982], and rely on two kinds of connections: symmetric links to encode attractor states, and slow asymmetric links to encode transitions between states. Sequence recognition in [Kleinfeld, 1986] is achieved by adding input units with connections to the memory units. Activation from these input units is required to complete transitions from one state to the next. These dynamical systems models can learn to represent and recognize simple sequences, but it is not clear how they could take advantage of the strong temporal structure in problems such as ours.

### Fixed Recurrent Links with Backpropagation

From its inception, researchers have attempted to extend the backpropagation algorithm [Werbos, 1974; Rumelhart *et al.*, 1986a; Parker, 1985] to learn and recognize sequences. Two early attempts use a set of "context" units in the input layer, at which fixed-weight links arrive from the hidden layer [Elman, 1988] or output layer [Jordan, 1986]. In the latter case, the context units also have individual fixed-weight self-links. Various versions of this type of backpropagation network have been shown to mimic finite state automata,

[3]It is possible to use delay lines to implement a buffer (e.g., [Tank and Hopfield, 1987; Waibel, 1989], but this is subject to all the problems of explicit unit buffers.

produce sequences or recognize sequences (e.g., [Cleermans *et al.*, 1989]). There is no explicit representation of time. Rather, the internal states which the network develops as it learns sequences code for temporal properties of the input; but as with other hidden unit representations, these encodings are usually opaque. Another type of network using context units [Mozer, 1989] allowed modifiable weights on the recurrent links, necessitating a new version of the backpropagation algorithm.

### Real-Time Recurrent Backpropagation

The backpropagation algorithm has been shown to be a special case of a more general gradient descent procedure that operates in networks with recurrent connections (recurrent backpropagation) [Pineda, 1987; Almeida, 1987; Rohwer and Forrest, 1987]. However, these derivations assumed fixed inputs and an approach to an attractor as the output. Several techniques for learning sequences in backpropagation networks with modifiable recurrent links have been developed [Mozer, 1989; Williams and Zipser, 1989; Pearlmutter, 1989; Fahlman, 1991]. These techniques can be used to learn sequences or stable limit cycles, and the continuous-time form has been extended (on paper) to allow time constants in the network update equations [Pearlmutter, 1990]. Although these formulations can be used to learn time-varying one-dimensional signals, it is not clear how they could be used in practice to learn time-varying imagery such as MLDs. Moreover, as with all implicit representations, it is hard to see how the learned sequence could be used to provide top-down feedback to the low-level feature processing that occurs in recognition of MLDs.

### Explicit Sequences

[Chun, 1986] outlines a representation for temporal sequence in massively parallel networks. His focus is the representation of temporal constraints, using single units to represent events and *constraint* units to mediate the links. The constraint units receive inhibiting inputs (link open unless active) and precondition inputs (link closed unless active). The units integrate arriving activation over time, subtracting a decay factor, and fire if a threshold is reached. The scheme can recognizes sequences of discrete events. Temporal duration of an active input can be represented and classified by cascading nodes (the integrative process takes time, so sequences of nodes take more time). These mechanisms are too crude for our task; in particular the scheme does not seem suitable for dealing with incoming sequences in which the initial phase is unknown, one of the important requirements in MLD recognition.

## 2.5.7 Hidden Markov Models

The representation developed in the next chapter has some similarities with Hidden Markov models [Rabiner and Juang, 1986; Bourlard and Wellekens, 1988]. HMMs have been used extensively and successfully for speech recognition tasks [Huang *et al.*, 1990], and gait

recognition is in many ways a visual analogue of the speech recognition task. We were unable to determine whether or not an HMM-based system would be suitable for the visual task, but it is clear that the solutions in the speech domain could not be applied without additional work. Two problems that would need to be addressed are the fact that multiple sequences are occurring simultaneously and in a coordinated fashion (arm-movements and leg-movements); and the fact that tempo changes are global - applying to all four limbs - and occur slowly. Investigation of an HMM-based solution for MLD recognition, particularly in comparison to the work presented here, would seem to be a worthwhile task.

# 3 Mechanisms

This chapter describes the different mechanisms that have been developed in the course of designing a network architecture capable of performing the recognition task. Recall that the major computational problems to be addressed are: representation and processing of sequence and time; indexing movement models with spatially-indexed features; formation of perceptual gestalts over space and time; and attention as applied to the latter two. In this chapter we discuss various solutions to these problems in the context of visual recognition of moving stimuli. For each solution, we discuss the trade-offs between functionality and efficiency that have to be made. The reader should mentally step back from the specifics of moving light displays and consider the abstract problems of dealing with sequence and time, indexing central representations with spatially-indexed time-varying features, and the formation of perceptual gestalts over space and time.

## 3.1 Representing Sequence and Time

The essential difference between recognizing a *walking* person and a static person is that the former involves integration of a time-varying input, while a single frame is sufficient for the latter. Recognition of time-varying input requires a representation of sequence and time. Although many kinds of time-varying input may be discriminated by sequence alone, in general we need to be able to discriminate on the basis of time as well (for example, strolling, walking, hurrying).

Suppose we have a set of feature units, presented with a time-varying input[1]. At each instant one of the feature units is active, but the active feature changes with time[2]. Suppose there are a number of canonical sequences of patterns of activation over the feature units, with a particular time trajectory associated with each sequence. We shall call these sequences and their time trajectories *scenarios*. Our task is to devise possible

[1]Throughout this work, we assume that the sequence is presented to the input units as it occurs; there is no input buffer.

[2]The presentation deals with patterns of activity in which at any instant only one unit is active, but this restriction is easily relaxed.

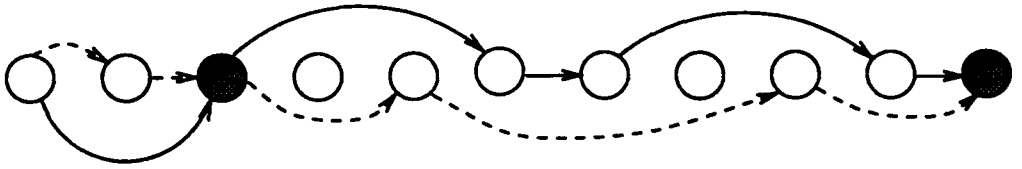

Figure 3.1: Link Representation of Sequence

representations for a scenario. This formulation is inherently discrete - the input consists of a sequence of distinct patterns of activation over the feature units, with clear delineation between the elements of the sequence.

First note that that a full representation must include aspects of both sequence - which pattern or element in the sequence follows which - and time - what are the time intervals between elements in the sequence. We may also want to be able to parameterize the representation in various ways - for example, scaling all the time intervals by a constant factor, or allowing a particular percentage variability in the time between adjacent elements of the sequence. We do not allow optional elements in a sequence (i.e., insertions and deletions), and shall not consider higher order interactions (e.g., timing constraints between three elements that do not degenerate into binary constraints between adjacent elements).

### 3.1.1 Representing Sequence

The simplest representation of sequence is shown in Figure 3.1. Here two sequences are shown, one using solid links, the other dashed links. If the links carry priming activation, then as a particular sequence progresses, activation will flow along one or both of the chains. Note that this representation is invariant with respect to translation in time - that is, the phase of sequence presentation is irrelevant (except that for acyclic sequences, less input will be seen as the phase difference between the canonical sequence and the input sequence grows). There are at least three other important aspects of this representation, that may be considered positive or negative attributes.

1. It is an implicit representation. If another processing mechanism needs to know when a particular chain is active, it cannot simply monitor a single unit or small group of units. There is no grandmother cell for the sequence as such.

2. As illustrated in the Figure, there is in general crosstalk between the chains. If either of the shaded units is highly active, and no other units in the chains are active, there is no representation of which of the two chains was in fact followed. Moreover, if the first shaded unit is active, it will prime both of the chains even if the history of activation was exclusively along one of the chains.

3. The strength of activation of a feature unit confounds two pieces of information - the degree to which lower-level input supports it; and the degree to which preceding elements of a chain in which it participates has been previously activated.

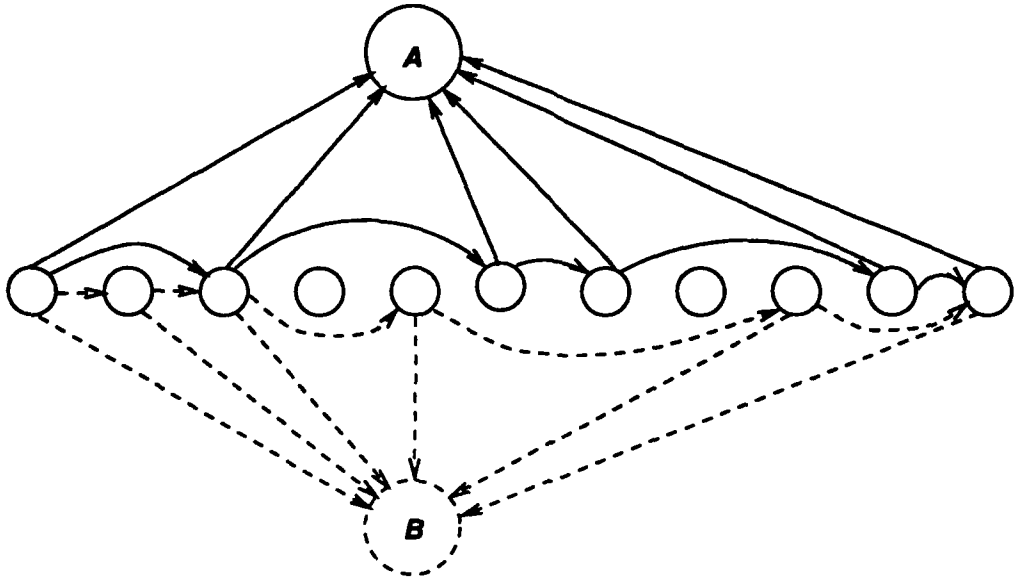


Figure 3.2: Summation Cells for Sequences

In general these three items should be considered deficiencies. However, this simple link representation has one major advantage: it is cheap. A complete sequence is represented by as many links as there are elements in the sequence. The remainder of this section explores variations on this representation that counteract one or more of these deficiencies.

### Implicit Representation and Crosstalk

The problem of implicit representation is common whenever the representation is distributed over many units. Here it is compounded by being distributed in time as well as space. The simplest solution is to augment the representation with a single unit for each sequence (a "grandmother" cell). The activity of this unit indicates the degree of belief that the sequence is occurring. Figure 3.2 illustrates this addition. The units labeled A and B receive inputs from each of the feature units that comprise their respective sequences. We call these "summation" units. If these inputs are undifferentiated, then the best the unit can do is to integrate the inputs over time. The more elements of the appropriate chain that occur, and the more strongly the corresponding feature units fire, the stronger will be the activity of the summation unit.

If the *order* in which the elements occur is incorrect, the summation unit will be less active because the feature units will not be appropriately primed. We can go further and give the summation units more internal complexity (i.e. state information). An ordered summation unit represents the order in which input is supposed to arrive, and adjusts its activity according to whether the order is followed.

The summation unit can be used to modulate the priming along the links that represent the sequence. This is shown in Figure 3.3. The feature units receive pairs of priming links.

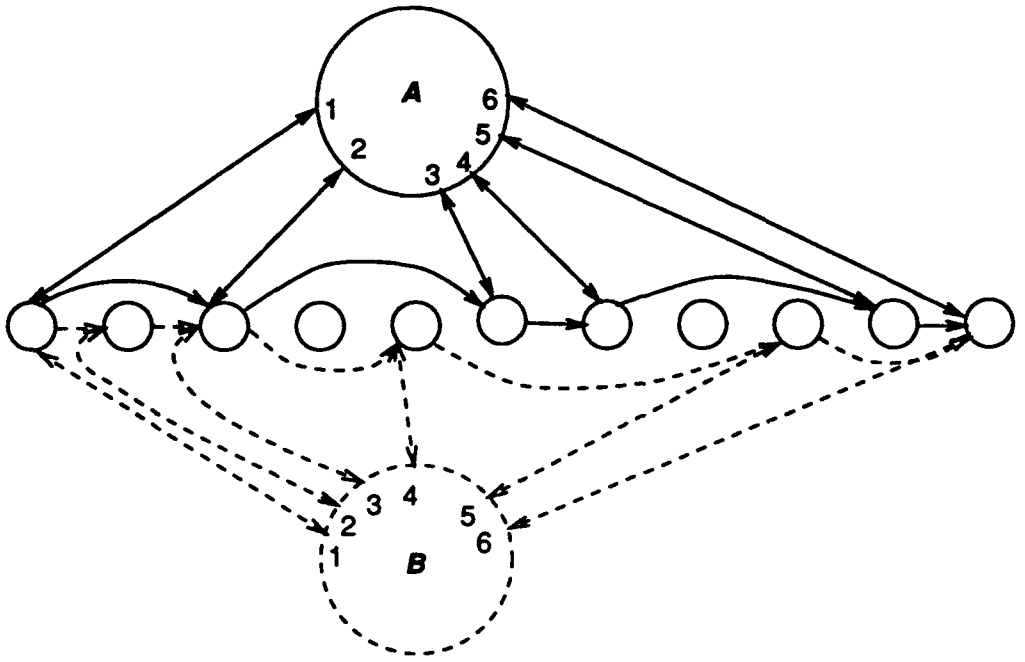


Figure 3.3: Combined Summation Unit and Link Chains

One link in each pair is from a summation unit; the other is from the preceding feature unit in the sequence represented by the summation unit. The feature unit combines low-level input with priming in a manner that is modulated by the input form the summation unit. One type of modulation is for the priming to be gated by the summation input. This restricts the priming to feature units in sequences whose summators are active, thus ameliorating the first two deficiencies listed above.

A second type of modulation is for the feature units to rely more heavily on lower-level input when the summator is low, and increasingly to require priming as summator input rises. This is particularly useful if we have a dedicated set of units to represent each sequence. It effectively implements a soft sequence constraint, with sequence checking becoming stricter as confidence in the sequence increases. Using this soft constraint, the summator can achieve low activation quickly without sacrificing precision sequence checking at higher levels of activation. We use these soft constraints in the scenario implementation (see section 3.1.3).

### Confounding Past and Present

A unit in the sequence representation shown in Figure 3.1 receives two kinds of activation: input activation from lower-level units; and priming activation from units which precede it in the chains of units that form the sequences. The unit's output is a scalar value that combines and confounds these two sources of activation. Combination is required for integration of input over time. Why would one want to make a distinction between activity

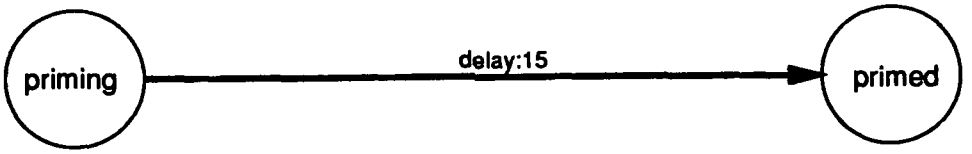


Figure 3.4: A Simple Delay Link

due to current input and activity due to integration over time? Consider the situation in which the units are arranged in a hierarchy. If the units that are formed into chains also provide input to higher levels, then the higher level units will be unable to distinguish to what extent the input they are receiving is due to the current state of the scene and to what extent it is due to the preceding activity in the scene.

This indivisibility of activation might be an advantage, to the extent that we can treat vision as controlled hallucination[3]. A prior input sequence that conforms to an expected sequence should influence the interpretation of current input. But that influence should not be sufficient to overrule input data except for very short intervals. The question is, how much should we allow information from the past to "leak" into the present [Freyd, 1983]. One solution would be to maintain two parallel hierarchies, one computing the features of the current input without reference to what has gone before, and the other accumulating information over time, using the one or more of the sequence mechanisms outlined above. An obvious problem with this idea is that it doubles the amount of hardware required, for uncertain gain. A deeper problem is that it is often the case that the current input is not sufficient to reliably compute any high-level features; temporal integration is required for disambiguation. An entirely different approach, which we shall take, is to treat this problem as just another aspect of the general perceptual problem of *gestalt* formation. In section 3.3 we discuss the problem of gestalt formation, and propose a solution.

### 3.1.2 Representing Time

We need to explore how the simple representation presented in Figure 3.1 can be extended to deal with actual time intervals between the occurrence of each element of the sequence[4]. A simple representation is to associate a time delay with each priming link, as shown in Figure 3.4. The delay on the link between two successive feature units in a chain is equal to the interval between the elements those units represent. In this scheme, the link acts very much like a biological axon: activation (a spike train) is fed in at one end and at some time later, it comes out at the other end (although conductance time constants are thought to be negligible relative to those of neuronal processes). The representational power of

[3] A delightful metaphor. The operative word here is "controlled"; hallucination is all too easily achieved [Thompson, 1971].

[4] We could concern ourselves with the duration of each element, or the time between onset of successive elements; these are equivalent. The presentation assumes the latter formulation.

the simple delay link is severely limited. We need to extend it to include the following information:

- We might wish to dynamically scale the delay using input from another network that computes the scaling factor.
- The delay might not represent a single point on the time dimension, but rather an interval; that is, the time period between two elements of a sequence might be acceptable if it is within some range. The extent of this interval of acceptable delays should also be dynamically adjustable.
- The knowledge that the priming unit shipped some activation down the link is lost until it reappears at the other end (i.e., for the duration of the delay associated with the link). We would like this information to be available to other parts of the network during the delay period.

If we extend the notion of what a link is, then these items can be dealt with. The scaling problem can only be dealt with by allowing the link to receive input that affects when it produces output, but that does not produce output by itself. The point to interval transformation can be achieved by making the output of the link a temporally smoothed (as well as delayed) version of its input, with the smoothing parameters provided by yet further inputs to the link. The "disappearance" problem can be addressed by having the link output a low level of activation immediately upon input, with that level rising after the appropriate delay. The link has become exceedingly complex.

Instead of endowing a link with all these computational and representational powers, it is better to use a unit which is dedicated to representing time interval parameters. The unit receives the priming activation, delays it according to a scaling factor, smoothes it according to a smoothing factor, and outputs the result to the unit that should be primed. It may output a low level of activation immediately if that is necessary. Figure 3.5 shows an example of two *event* units connected by an *interval* unit[5]. The interval unit's transfer function is shown to the right of the network. This depiction of the transfer function is intended to be representative of a class of functions. The essential characteristics are detection of rapid onset of activation in the event unit, then setting the output to be at or just above resting potential until near the appropriate interval, after which the output rises, reaching a maximum at exactly the correct interval; then the output dies down to resting potential. The window of activation allows for some deviation from the expected interval. It is important to note that the interval unit is not a delay line: it detects a rapid increase in event unit activation, starts its clock, and later outputs the activation as priming to the next event unit. The rapid increase in event unit activation is taken as a signal of when the event

[5]Henceforth, we shall call units that represent the elements of a sequence event units, denoted in illustrations by circular icons. Units that represent time intervals between elements of a sequence shall be called interval units, and denoted in illustrations by non-circular elliptical icons.

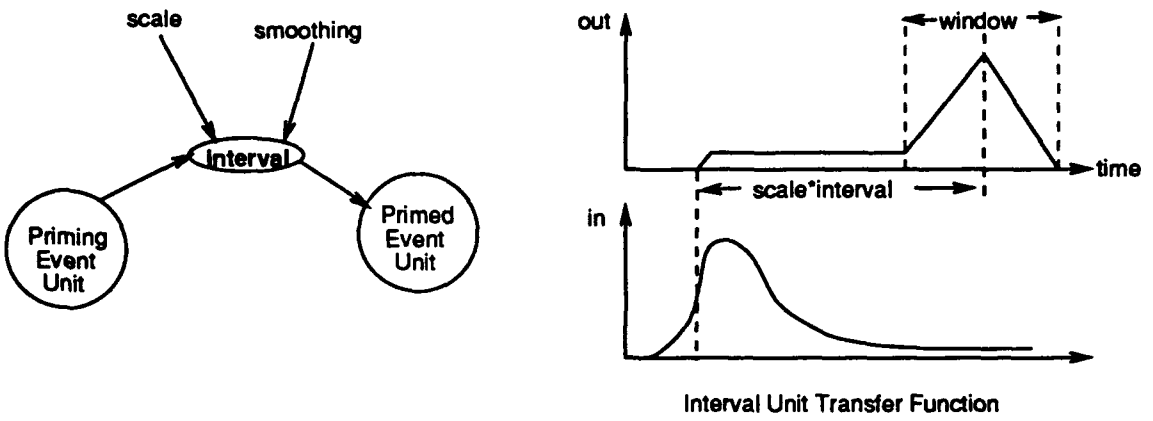


Figure 3.5: A Simple Interval Unit

occurred, which is necessary in order to model inter-event time intervals. This is the basic representation of time adopted in this thesis.

A particular type of clock mechanism can be provided by an *integration* unit. This type of unit sends activation to itself, with the weight on the self-link controlling how fast activation increases. A linear threshold unit connected to the integration unit will fire after a delay depending on the threshold and the rate at which the integration unit's activation increases. A range of time intervals can be implemented with low (start) and high (stop) thresholds. This is essentially the mechanism adopted in our interval units, but it is all contained in one unit. Two other ideas for clocks are presented in [Olson, 1989], and it seems clear that there are many computationally equivalent ways to implement the basic idea.

### Time Interval Ranges

A more flexible form of interval unit represents a range of time intervals rather than a single-valued time interval. This type of interval unit is parameterized by the shortest and longest expected time intervals, a tempo factor, a strictness factor, and two threshold factors.. Its function is to respond to the onset of the prior event unit and provide output during a temporal window determined by its parameters. We call this window the *expectancy window* of the interval unit. Figure 3.6 shows input/output plots for this type of interval unit. The range of time intervals is shown by the high output plateau, the strictness factor controls the slope of the ramp up to and down from the plateau, and the tempo parameter scales the canonical time intervals. We discuss ideas for how the tempo parameter could be determined and what would be reasonable limits on its values for our domain in Chapter 7. The strictness parameter is dynamically determined; we discuss this below. The threshold factors determine when the interval unit starts or restarts its internal clock, as described below.

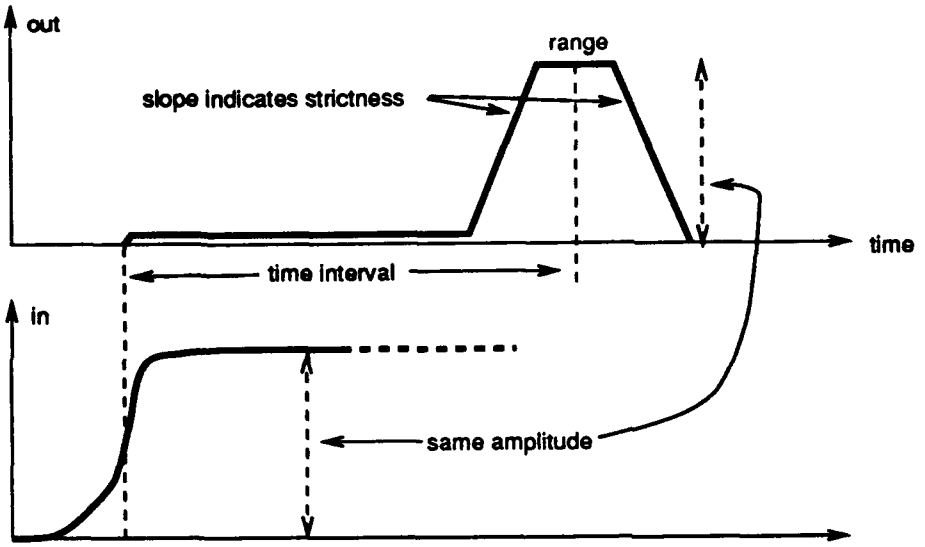


Figure 3.6: Interval Unit Transfer Function

The unit function $P(t)$ for the interval unit is a multi-state function, with one state for each of the linear segments shown in Figure 3.7. The state names are shown in italics. During normal operation the states follow each other as shown in the Figure, under the control of the unit clock, with two exceptions: 1) The transition into the *counting* state is called enablement and is controlled by rate of change of the event unit activation 2) during any state except *quiescent*, the unit may be re-enabled, causing the state to switch to *counting* and resetting the unit clock. In addition to the unit state and clock, the unit maintains one other state variable, the *reference* value, which is the maximum activation attained by the event unit since *reference* was last reset. Denoting the event unit activation at time $t$ by $E(t)$, then if at time $t$ the unit is in the following state, then the unit applies the following condition/action rules:

- *Quiescent.* If $E(t)/E(t-1) > \alpha$ then enable: *clock* $\leftarrow 0$, set *reference* $\leftarrow E(t)$, and *state* $\leftarrow$ *counting*.
- *Counting.* If $E(t)/E(t-1) > \beta$ and $E(t) >$ *reference* then re-enable: *clock* $\leftarrow 0$, reset *reference* $\leftarrow E(t)$, and *state* $\leftarrow$ *counting*. Otherwise, if the *clock* has reached the end of the *counting* period, then *state* $\leftarrow$ *increasing*.
- *Increasing.* If $E(t)/E(t-1) > \beta$ and $E(t) >$ *reference* then re-enable. Otherwise, if the *clock* has reached the end of the *increasing* period, then *state* $\leftarrow$ *plateau*.
- *Plateau.* If $E(t)/E(t-1) > \beta$ and $E(t) >$ *reference* then re-enable. Otherwise, if the *clock* has reached the end of the *plateau* period, then *state* $\leftarrow$ *decreasing*.
- *Decreasing.* If $E(t)/E(t-1) > \beta$ and $E(t) >$ *reference* then re-enable. Otherwise, if the *clock* has reached the end of the *decreasing* period, then *state* $\leftarrow$ *quiescent*.

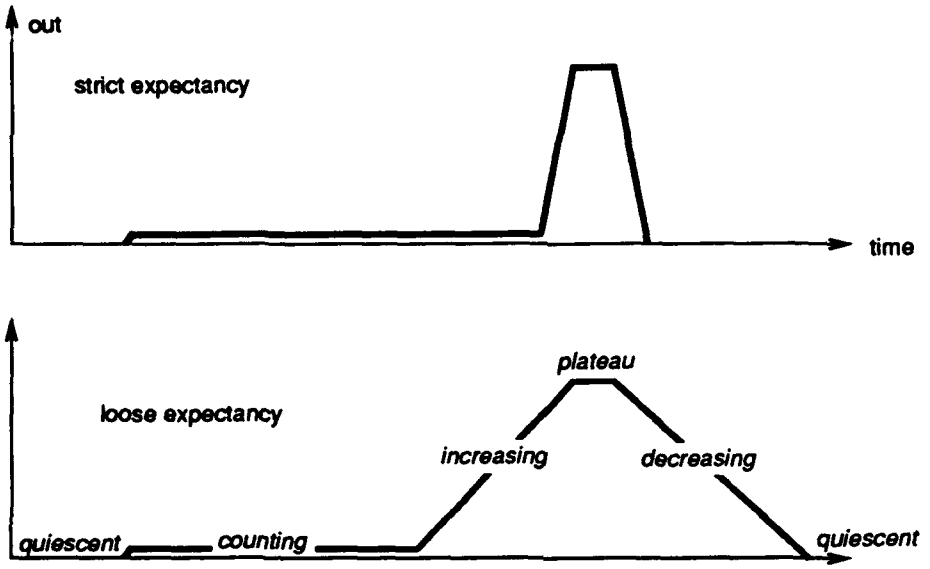


Figure 3.7: Loose and Strict Expectancy Windows

The parameters $\alpha$ and $\beta$ are control what spike in activation level from the prior event unit is required to start or restart the interval unit's clock. Their setting depends on the form of the activation function in the event units.

### 3.1.3 Representing Scenarios

We use the ideas developed in the preceding sections to represent scenarios. In its most general form, the scenario is a structure for representing a discrete sequence of events separated in time by specific intervals or ranges of intervals. It is defined by a set of events, a complete temporal ordering on those events, and a specification of the intervals between consecutive events. An event is characterized by a set of *supporting* conditions and one or more *enabling* conditions. The supporting conditions specify the lower-level features that are active between the onset of the event and the onset of the succeeding event. The enabling conditions specify a change in lower-level feature activation that indicates the event has commenced. Enabling conditions are required to locate the time of occurrence of an event explicitly, which is crucial for implementing precise temporal constraints in the interval units. An interval is characterized by a range of time delays exactly as described in the preceding section on interval units.

A fragment of the scenario representation is shown in Figure 3.8. It uses one unit for each event and one unit for each interval. The event units perform sequence checking and the interval units perform time interval checking. The Figure shows the representation of two consecutive events in a scenario, *A* and *B*, and the interval between them. Focusing on event *B*, we see that priming activation is provided by the previous interval unit, according to the function illustrated in Figure 3.6. Enabling and support conditions are provided by

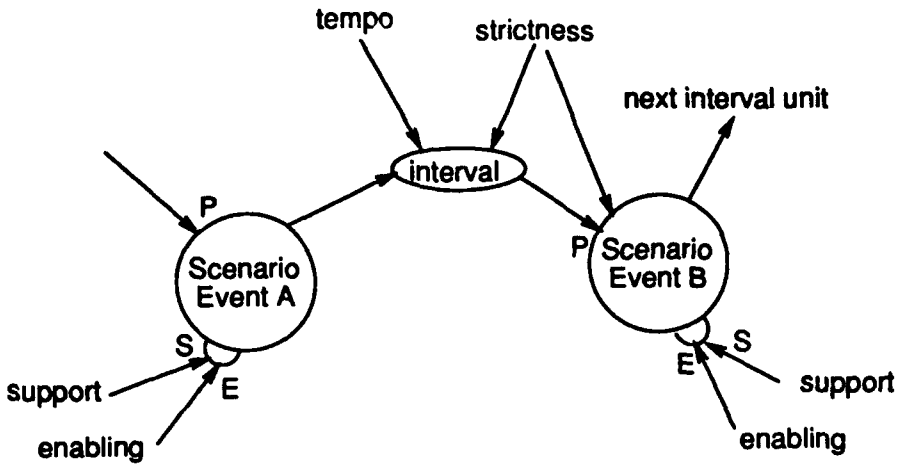


Figure 3.8: Scenario Representation

lower-level features. The strictness factor is provided by a summator unit described below and is used to control the strictness with which sequence checking is applied. An event unit combines activation from its support features, from detection of its enabling conditions and from the priming interval units, together with the strictness factor. In the absence of any priming from prior interval units, the event unit simply sums up its support. If an enabling condition has occurred, the event unit boosts the activation due to support features by a significant factor. This spike in activation serves to indicate the time at which the event occurred. Enabling conditions are discussed further in section 4.2.1. Priming activation from prior interval units acts to increase the activation of the event unit. The activation function is:

$$E(t) = \alpha S(t)(2.0 - X(t)) + \frac{\beta P(t) + \gamma\sqrt{P(t)S(t)}}{2.0 - X(t)}$$

where $P(t)$ is the priming provided by prior interval units, $S(t)$ is the support provided by the support conditions modified by the boosting factor due to enabling conditions being satisfied (if any), X(t) is the strictness factor, and $\alpha + \beta + \gamma = 1$. The strictness factor is in the range[0, 1]. When it is low, input from lower levels, i.e. $S(t)$, is relatively more significant than when the strictness factor is high. When the strictness factor is high, priming $P(t)$ is relatively more significant. As strictness increases, the event unit concentrates more on priming activation which indicates correct sequence.

Figure 3.9 shows an activity trace for the three units in Figure 3.8. The horizontal axis in each panel depicts time, the vertical axis activation. In the left-hand panel, we see the enablement of the unit representing event A, indicated by the spike in activation. This starts the interval unit's clock, which then controls the expectancy window of the interval unit, as seen in the middle panel. During the plateau phase, the unit representing event B is enabled, as indicated by the activation spike in the right-hand panel. The B unit reaches a higher level of activation than the A unit because of the priming from the interval unit.

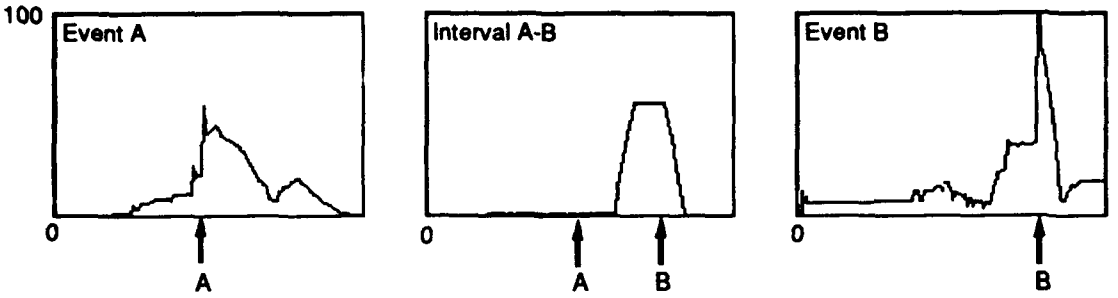


Figure 3.9: Activity Traces

### Scenario Summator Cells

The summation units described in section 3.1.1 are added to the scenario network to form evaluator cells representing the whole scenario. Each summator represents the overall belief that the scenario is occurring. It is desirable for the temporal and sequence constraints expressed in the scenario to be lax when there is much uncertainty as to the interpretation of the scene, but to become progressively tighter as the interpretation becomes firmer. We implement this by using a summation unit output to control the strictness in the event units (which check sequence) and the interval units (which check time intervals). Initially, when the scenarios have little activity (uncertain interpretation) and the strictness parameter is low. This causes event units to concentrate on lower-level input and interval units to have broad expectancy windows. As a scenario becomes more active and its summation unit reflects this, the event units focus more on priming and the interval units have narrower expectancy windows. Figure 3.7 illustrates broad and narrow expectancy windows.

The summation unit monitors the activity in the scenario event and interval units to provide an estimate of overall scenario activity. We use a summator unit with a separate site for each event. The site receives links from the event unit and the *subsequent* interval unit. Normally, the site maintains the maximum activation it has received from the event unit. But if the activation arriving from the interval unit is decreasing, this means that the expectancy window is terminating, so the site activation declines to zero. The result is that each site is active when its event is active, the degree of activation indicating the strength the event achieved during its lifetime; and end of the event's lifetime is indicated by interval unit. In a scenario with symmetry (e.g., *pair-of-legs-walking*, in which left and right legs are indistinguishable in 2D), there should be two active sites at any time if the scenario models the input. Therefore, the summator unit takes as its activation the average of the two most active sites. The output of the summation unit is an exponential approximation to that activation, to avoid discontinuities in output levels.

Figure 3.10 illustrates a four-event scenario, showing the summator unit and the links between the event, interval and summator units. Each site on the summator units receives links from one event unit and the subsequent interval unit. The summator unit sends activation back to the interval units to control the strictness of the expectancy window, and

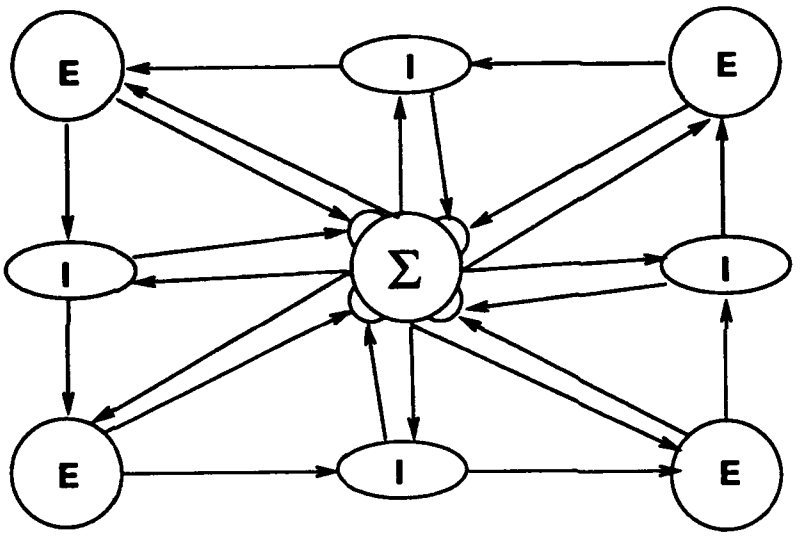


Figure 3.10: Summator Links

to the the event units to control the strictness with which sequence constraints are applied.

## 3.2 Activating Scenario Representations

Representations are not sufficient unto themselves. Representations involved in perceptual and motor functions are always invented or learned as an intermediate form for one of two purposes: recognition or generation[6]. We are interested in recognition. The representation should accept input and become active in a manner that reflects how the input matches *whatever is represented*. Whether different items are represented by different sets of units, or distributed over a common set of units, the task is to ensure that only the correct, or nearest, set of units becomes active.

Consider the sequence/interval representation illustrated in Figure 3.11 in the context of recognition. The sequence represented by such an assembly of event and interval units is a scenario[7]. Over time, input activation I will be arriving at the event units (circles), bottom-up or top-down or both. *If the sequence and timing of the incoming activation* corresponds to the scenario, then a peak of activation will flow around the cyclic network. Because of the priming one event provides to the next, the peak activation will be much enhanced if the scenario matches the input. In this case, we say the scenario resonates with the input.

[6]Organisms interact with their environment; all internal representations are intermediate in the sense that they appear between sensory input and motor output. A representation may serve both recognition and generation purposes at once.

[7]Many natural sequences are cyclic, e.g., gait patterns, but the analysis presented here does not depend on scenarios being cyclic.

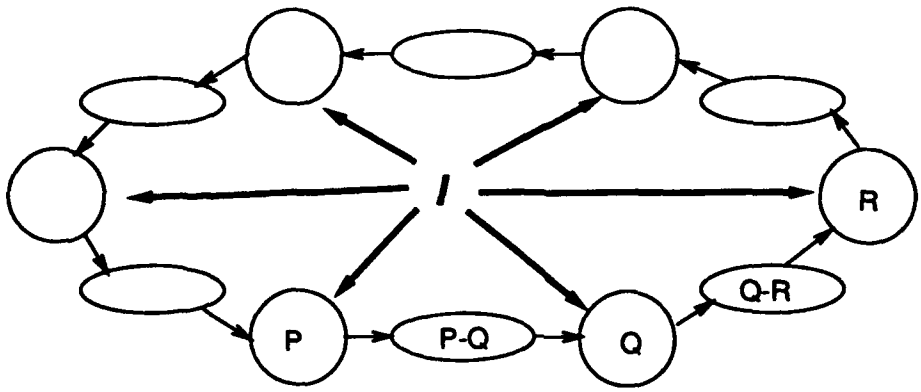


Figure 3.11: A Scenario

### 3.2.1 Crosstalk

The detailed design must be sensitive to correct inputs, but also robust against possible confusions. There are two ways in which resonance can be disrupted, even when an exact match occurs:

- Multiple input sequences are presented to the representation simultaneously. The different sequences contain common elements or subsequences.
- The sequence contains multiple copies of the same element or subsequence.

In both cases, the problem is one of crosstalk: in the first instance because of crosstalk between similar elements of different sequences; in the second, crosstalk between similar elements of a single sequence. Referring to Figure 3.11, suppose the event unit P has fired, initiating the delay function implemented by the interval unit P-Q. Now what should be the response if P receives new input, before the succeeding event unit Q has fired? This could occur if two identical sequences are presented simultaneously, with events occurring in the order:

$$p_1 p_2 q_1 q_2 \ldots$$ [8]

or if a single represented sequence contains the subsequence:

PP

There is a problem because of the interval unit P-Q, which represents the fact that the relative times of occurrence of the events in the input is important. The question is,

[8] We use upper case to refer to units and the events and intervals they represent, and lower case to refer to input events and intervals. Subscripts on input events denote the sequence which generated them.

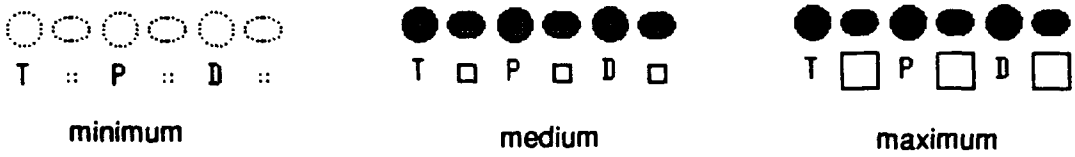


Figure 3.12: Scenario Unit Icons for different Activation Levels

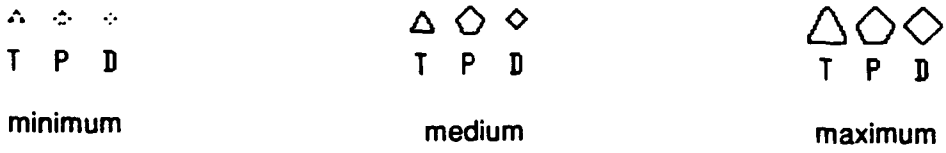


Figure 3.13: Feature Unit Icons

should the interval unit clock be re-initiated, or should the second input to P be ignored? The answer depends upon the overall behavior desired. First we deal with the multiple sequence case, which in the context of moving light displays could arise if, e.g., there are two walking figures. In this case, one of the two sequences must dominate the scenario, to avoid interference between them.

### A Network Simulation

In this section we present a network simulation of the multiple sequence case. Figure 3.12 illustrates the icons used to show what is occurring in the scenario units. The Figure shows three levels of activation for the different types of icons. Circular icons represent the activation levels of the event units. Elliptical icons represent the activation levels of the interval units. Below each event icon is a letter, indicating the type of feature which the event detects. Below each interval icon is a square icon indicating the state of the interval unit clock. For these icons, which show internal state of the unit rather than its activation level, the minimum level means the clock is not running, the maximum level means the clock has incremented up to the delay represented in the interval unit, and intervening levels indicate what proportion of that delay has elapsed since the clock was initiated. The icons represent the sequence progression from left to right, so that the Figure shows a scenario which consists of the event sequence TPD. Although not shown in the Figure, the time intervals between the events is 4 simulation steps in the network runs described below.

Figure 3.13 illustrates the icons used to show the occurrence of features in the input. As in the previous Figure, three levels of activation are shown for the three types of features. Feature T is shown with a Triangular icon, P with a Pentagonal icon, and D with a Diamond shaped icon.

#### Single Sequence

First let us examine what happens when the sequence TPD occurs in the input at the correct time intervals. Figure 3.14 shows how the scenario units respond to the input. The

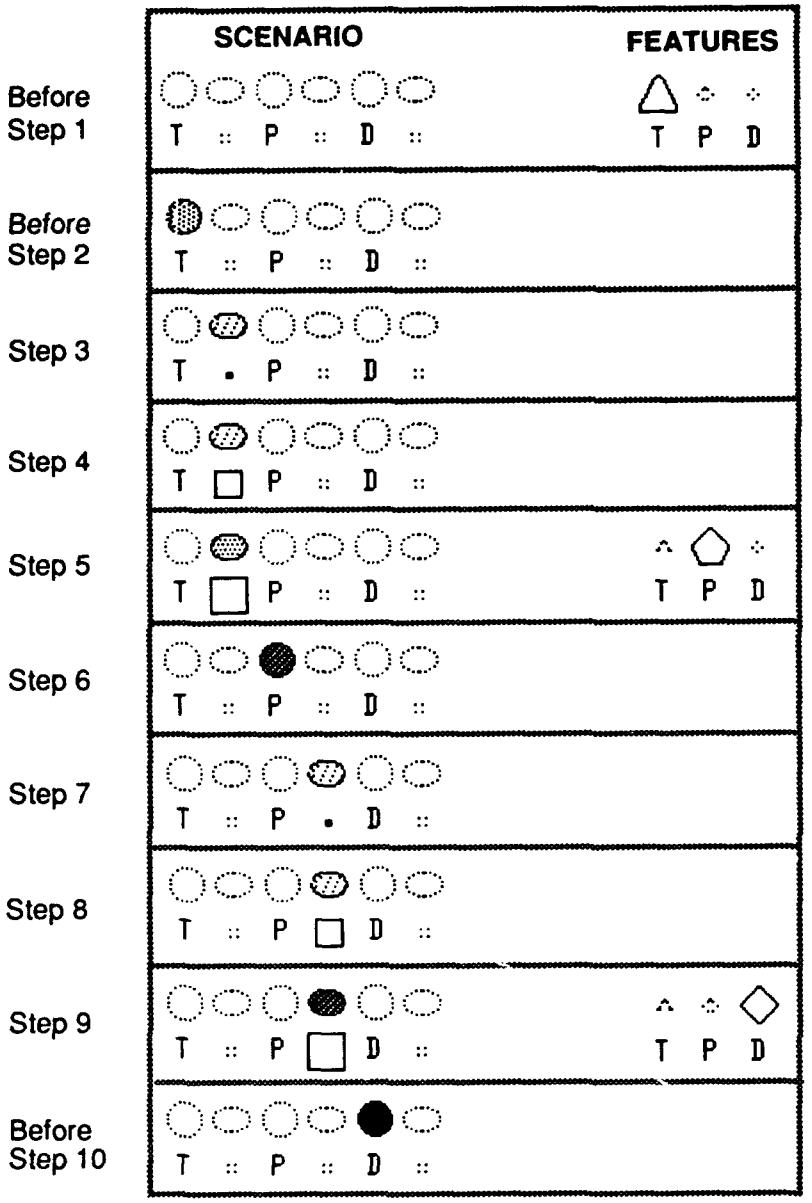


Figure 3.14: Single Sequence Presentation

left hand column shows the scenario units at each time step, with the initial situation shown at the top and the final situation at the bottom. The right hand column shows the input feature units when a feature occurs in the input. At other times, the feature units have zero activation. At time step 1, feature T is active, causing the T event unit to become slightly active at time step 2. The interval unit succeeding the T event unit starts its clock at time step 3, and has a very low level of activation while the clock is running. The clock increments at time step 4 and has reached its maximum level at time step 5, so that the interval unit sends priming activation to the next event unit, P. The priming activation is at

the same level as the interval unit received from the preceding event unit T, when it fired at time step 1. At the same time as the interval unit is providing priming to event unit P, feature P occurs in the input (step 5). Thus at step 6 event unit P fires strongly, since it received both priming and feature input. The reader can follow the input through steps 7, 8 and 9, during which time the next interval unit's clock is initiated and incremented. At time step 9 the clock has reached the correct delay value, so the interval unit provides strong priming to event unit D. At the same time, feature D occurs in the input. Thus, at step 10, event unit D fires at maximum, having received strong priming and strong feature input.

**Multiple Sequences**

Now consider the multiple sequence case, for the sequence TPD: if there are two instances of the same sequence, slightly out of phase, then we could have the input sequence:

$$t_1 t_2 p_1 p_2 d_1 d_2 \ldots$$

If the desired behavior is that the later sequence dominate, then the interval unit should be set to re-initiate itself whenever the preceding event unit fires. Let us examine a network simulation of this situation in detail. Figure 3.15 shows a sample network run. The Figure shows the scenario units on the left and sets of feature units at two locations #1 and #2, in the middle and on the right. The sequence TPD is presented in the feature units at location #1 at steps 1, 5, and 9. The same sequence is presented in the feature units at location #2, but two time steps later (steps 3, 7, and 11). The sequence of steps is shown in the figure from top to bottom. By running an eye down each of the feature columns for #1 and #2, it can be verified that each of these sets of units produce the correct sequence, with #2 later in phase than #1. The response of the scenario units to this input appears in the left hand column of icons.

Initially all scenario units have zero activation, and location #1 has feature T active. At the next step the event unit sensitive to T to become somewhat active. It has responded to the input from location #1.

One time step later (step 3), the T-P interval unit has received activation from the event unit and initialized its clock (the tiny square icon). The interval unit also becomes very slightly active (shown by very light shading in the ellipse icon). At this time the T feature occurs at location #2 (the large triangular icon).

At time step 4, the interval unit's clock has incremented (the square icon is larger), and the event unit sensitive to T has responded again (lightly shaded circular icon), this time to the input from location #2. The icons for time step 5 show the critical change. The interval unit which was already active, has reset its clock because of the second firing of the T event unit. Compare this with the clock state shown in step 5 of Figure 3.14. This means that the interval unit will now not provide priming to the next event unit (P). Concurrently, at step 5, the P feature at location #1 occurs. At step 6 the P event unit responds, *but weakly* because of the lack of priming.

At step 7, the interval unit's clock has reached its threshold so the unit (shaded more) provides priming to the P event unit. At the same time, the P feature occurs at location #2,

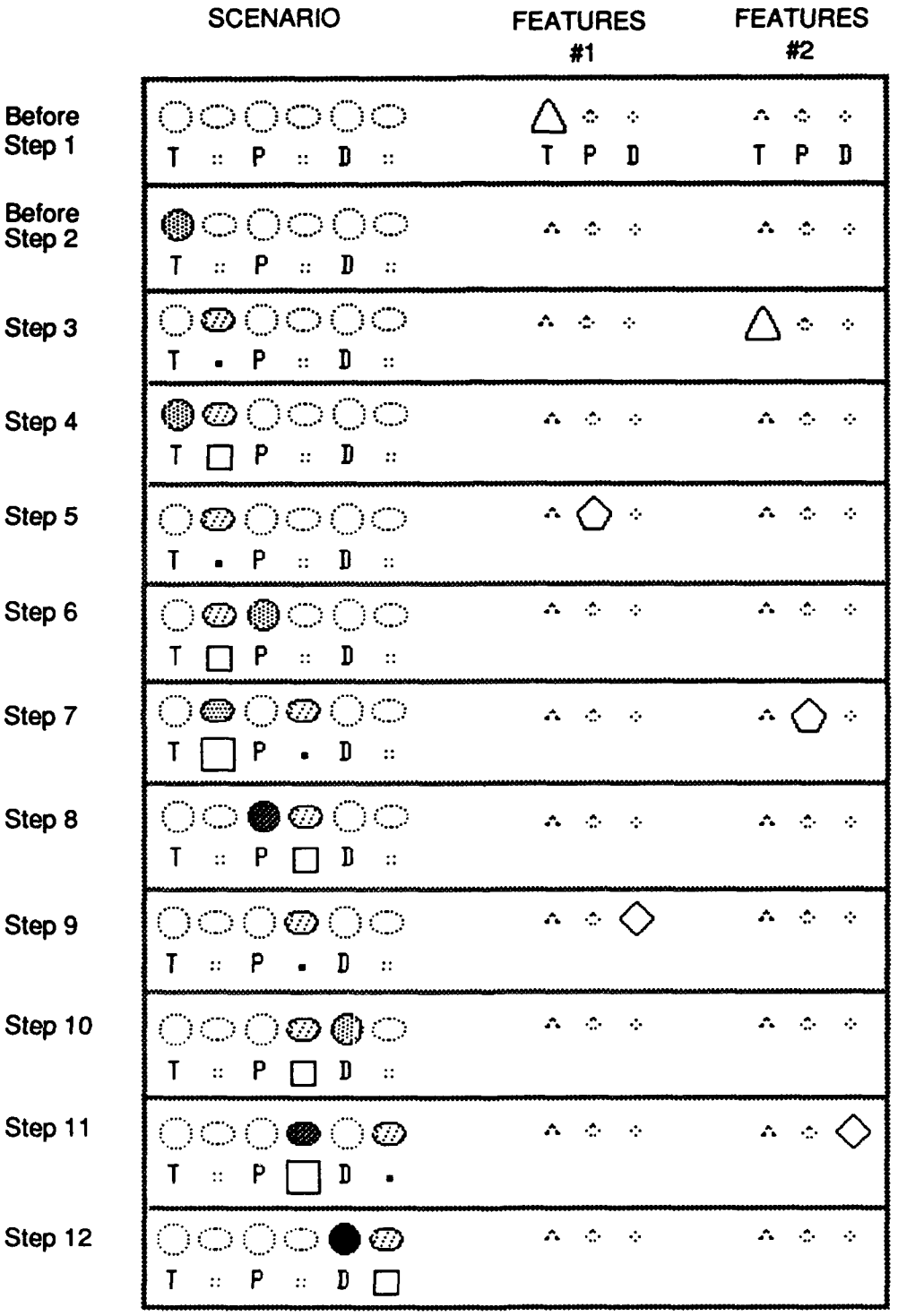


Figure 3.15: Multiple Sequence Presentation

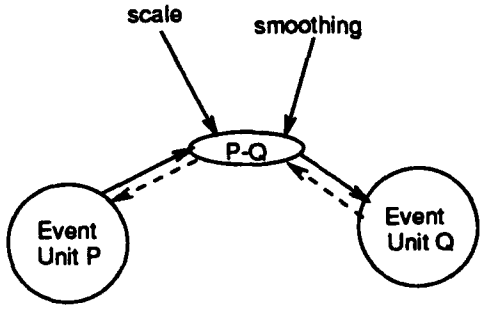


Figure 3.16: Reset Links

so that at step 8 the P event unit responds strongly, since it received both feature input and priming. As can be seen by the shading, the P event unit responded more strongly to the P feature at location #2 than it did to the same feature at location #1. The reader can follow the sequence to the bottom of the illustration. Notice that the response of the D event unit to the first D feature (location #1, step 10) is at the same level as the P event unit response to the first P feature (step 2); but the D event unit response to the later D feature (location #2, step 12) is at the maximum value. In summary, the network selects the later sequence.

If, on the other hand, the desired behavior is that the network select the earlier sequence, then the event unit should not respond to input until the succeeding event occurs. This could be achieved by endowing the event unit with a refractory period during which it does not respond to input. The length of the refractory period must correspond to the the time interval between the event and its successor. There are three problems with representing the refractory period directly in the event unit:

- The time period is already represented in the interval unit, which allows scaling and smoothing. We do not want to duplicate this representation.
- If a different event unit (B) responds more strongly to the input than this event unit (A) because, for example, B is more strongly primed than A, then unit A should remain sensitive to new input under the assumption that the current input is better assigned to unit B.
- If the interval between the events occurring in the input is shorter than the refractory period, then the event unit will still be in its refractory state after the succeeding event unit has fired. We need to be able to terminate the refractory state prematurely if this happens.

One simple way to provide the event unit with a refractory period while avoiding these problems is to provide two reset links. Figure 3.16 illustrates these links. The Figure shows event units P and Q with their associated interval unit P-Q. The normal priming links are drawn with solid lines, the reset links with dashed lines. One of the reset links is from the interval unit P-Q to the preceding event unit, P. When the activation arriving along this link

falls to resting potential, the event unit becomes sensitive to input again. This reset link solves the first problem outlined above, by using the interval unit to control the refractory period in the preceding event unit. The second problem requires a mechanism for mediating between competing event units. We defer description of this mechanism to the end of the section. The third problem is solved with the reset link from the primed event unit, Q, back to the interval unit P-Q. When event unit Q fires, activation flows along this link to the interval unit P-Q, causing the interval unit to reset into its resting state; which in turn causes the termination of P's refractory period.

Whether or not the reset links should be used is determined by the desired behavior of the scenario network. In the implementation we do not use reset links and in the following discussion and Figures they are omitted.

### Ambiguous Sequences

Crosstalk between similar elements of a single sequence is less of a problem, because the sequence is presented in order, so the similar elements are not presented simultaneously. As noted above, the network representation is naturally shift invariant (that is, shift with respect to time). Figure 3.17 illustrates the flow of activation when the network representing a cyclic scenario PPQRPQ, containing multiple common elements and subsequences, is presented with part of the scenario as input. Here all the inter-event time intervals are assumed to be equal[9]. In order, from top to bottom, the networks show the response to the input sequence *ppqrp*. When the first *p* arrives (top left), all three P units respond, weakly, as shown by the shading. When the second *p* arrives, two of the P units respond weakly because they receive no priming, but the P unit which receives priming (from the shaded interval unit) responds more strongly. When the *q* arrives, both Q units are receiving priming, but one is receiving weak priming, the other strong priming (note the shaded interval units). So one responds more strongly than the other; the correct one in fact. When the *r* arrives, the R unit is strongly primed and responds strongly. When the final *p* arrives (bottom left), two of the P units respond weakly, but the correct P unit responds strongly. Contrast this with the response to the arrival of the first *p*.

But Figure 3.17 illustrates a potential problem. Notice that when *q* is presented to the network, both Q units respond strongly, albeit that the correct one is stronger than the other. This is because both receive some priming. We can arrange that only one unit is primed in response to any particular input event, by adding an inhibitory site to each interval unit. This site receives input from each of the event units that are of the same type as its precursor (i.e., a P-Q interval unit is inhibited by all event units of type P, except its immediate precursor). The interval unit is initialized only if its excitatory input (from its precursor) is greater than or equal to the maximum inhibitory input. A fragment of the corresponding network

[9]Equal time intervals makes the problem harder, since each time interval may be confused with any other. With different time intervals, the crosstalk problems are less severe because the time intervals provide a basis for distinguishing correct and incorrect matches.

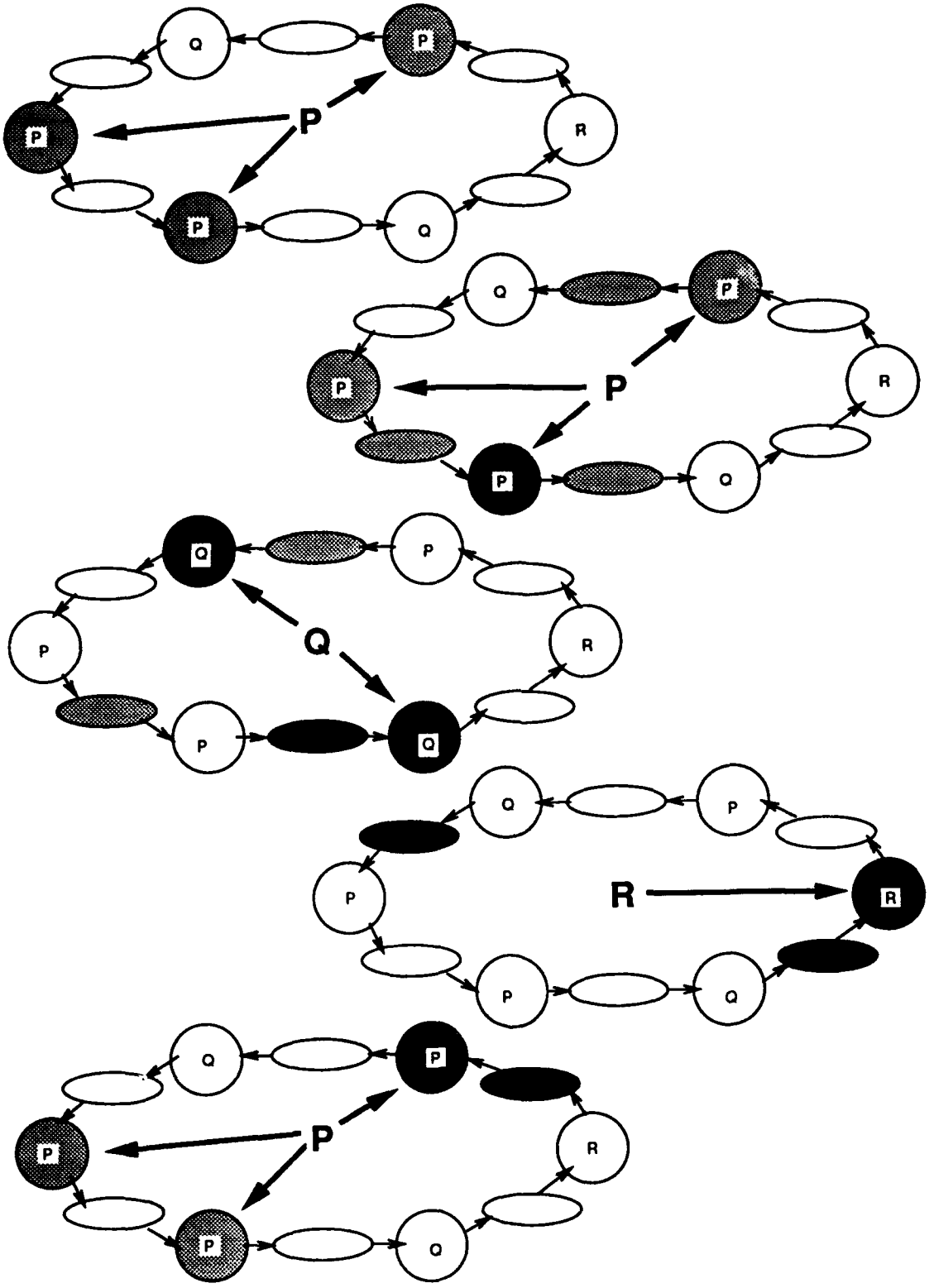


Figure 3.17: Shift Invariance

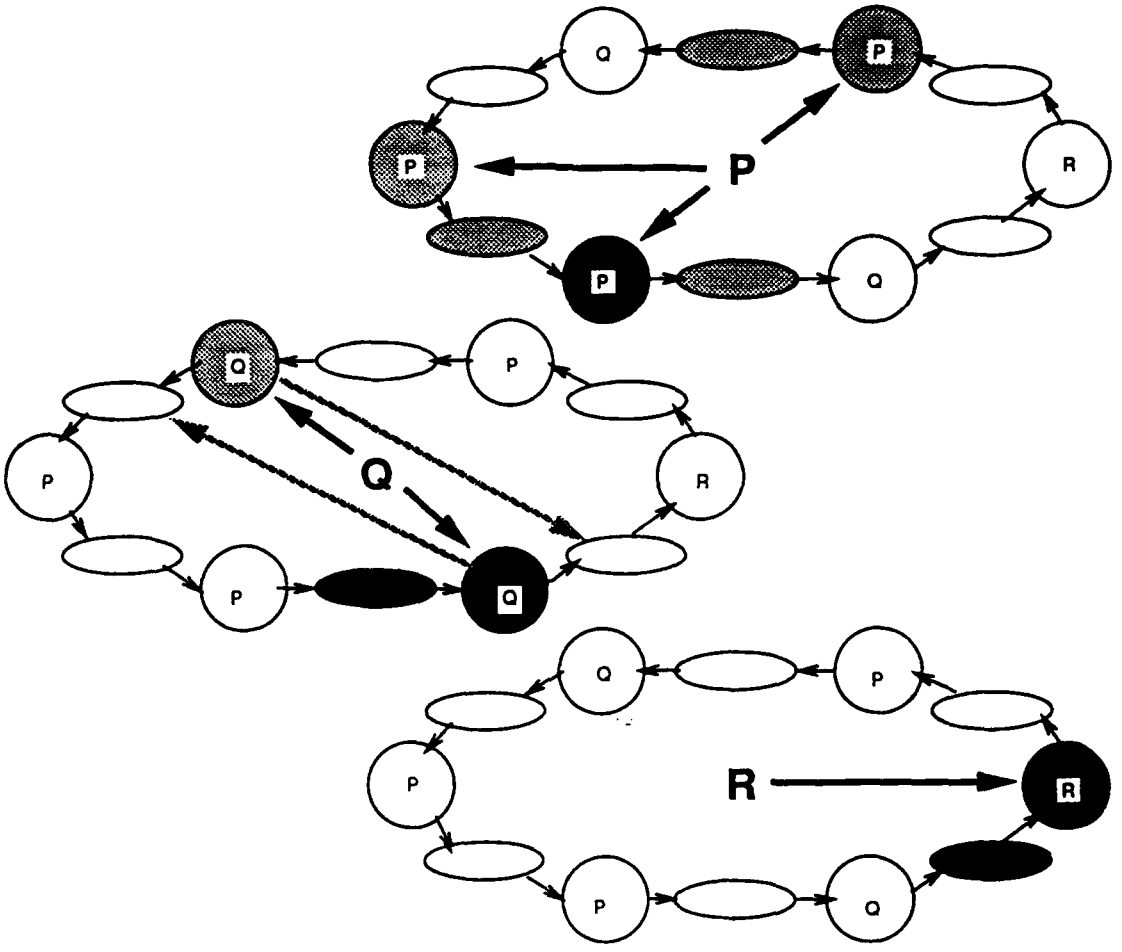


Figure 3.18: Winner-Takes-All Interval Units

operation is shown in Figure 3.18, which also shows the inhibitory links (shaded) from the Q units. For input *ppqrp*, Figure 3.18 shows the response to the second, third and fourth elements, i.e., *..pqr...*

One could imagine a scenario representation in which multiple copies of P, for example, were collapsed into one unit, connected via interval units to each of the allowable next events. Figure 3.19 illustrates such a representation for the sequences illustrated in Figures 3.17 and 3.18. In this case we have three event units, one each for P, Q and R, and six interval units representing the same time intervals as those in the previous figures. This representation is more compact, but it would be unable to discriminate between the inputs *ppqrp* and *pqpqrp*. We do not explore this kind of representation further.

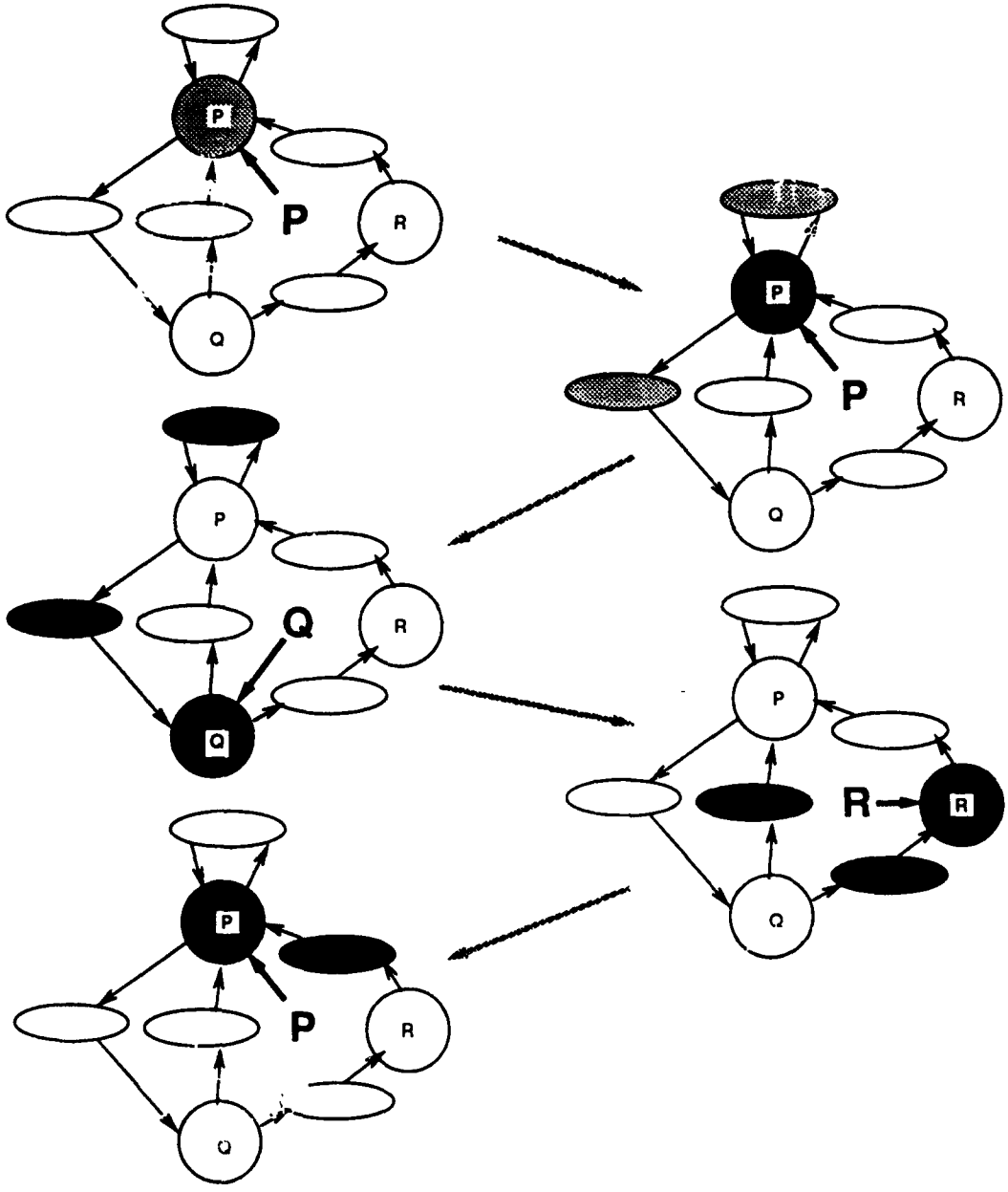


Figure 3.19: Multiple Use Event Units

### 3.2.2 Integrating Late Evidence

Suppose that at some time soon after the occurrence of the input pattern corresponding to an event, further evidence confirming the event arrives from some source. One way in which this can happen in the case of visual sequences is if the tempo of the input data is faster than that modeled in the scenario. Then the priming signal to the event may be increasing even after the onset of the event. This is a form of confirming evidence. It would be incorrect

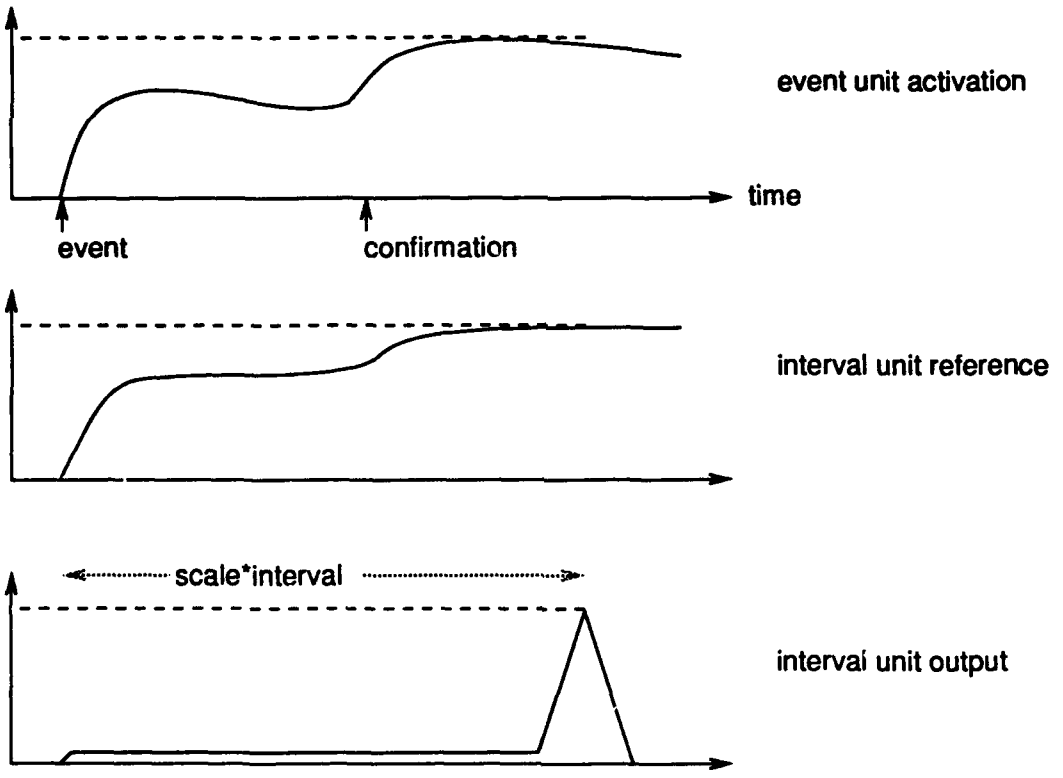


Figure 3.20: Responding to Increased Activation

for the event unit to ignore the confirming evidence, and it would likewise be incorrect for the event unit to treat the confirming evidence as a signal for the onset of the event. The correct behavior is to augment the activation of the event unit. But the interval unit, as described above, treats the onset of activation in the event unit as a signal to start its clock. We can integrate late evidence by specifying that the interval unit will only start its clock if the activation of the event unit increased sufficiently rapidly; otherwise it augments its reference activation (the value passed on to the primed unit) to the new level. This process is illustrated in Figure 3.20

## 3.3 Gestalt Formation and Attention

As recognition proceeds, the pattern of activation over the units should represent a coherent percept, with each detected feature assigned to a part of the perceptual construct. That this should occur manifests itself in human perception by our spontaneous organization of a scene into a single interpretation. This occurs despite widespread local ambiguity in the input data. The name given to the phenomenon is "gestalt perception". The various definitions of the term are discussed in [Olson, 1989]. We provide a short review here. The essential properties of a gestalt are *discreteness* and *consistency*. Discreteness refers

to the requirement that a visual feature is either part of or not part of the gestalt; as Olson says, there is no middle ground. Consistency refers to the organization of the gestalt, i.e., which features are assigned to which roles in the perceptual construct. A consistent construct is one in which a complex of constraints between roles and features is satisfied. The constraints derive from the physical nature of the environment, e.g., the three angles of a triangle add up to 180 degrees; and from knowledge about things in the world, e.g., a quadruped has two pairs of legs, one in front of the other. These constraints may be hard in the sense that they *must* be satisfied, or soft, in the sense that the relation represented by the constraint is *likely* to be true.

There have been several kinds of computational formulations of the gestalt idea, reviewed in [Olson, 1989]. They include Hough transform networks [Ballard, 1984], stable coalitions [Feldman and Ballard, 1982; Rumelhart *et al.*, 1986b], Harmony theory [Smolensky, 1986], Markov random fields [Cooper, 1989] and feature binding [Califano *et al.*, 1989; Olson, 1989]. We follow Olson's approach [Olson, 1989], which is to treat the formation of gestalts as a problem of binding features to explanations. He outlined a mechanism for binding low-level features to high level explanations in Hough transform networks, and a related mechanism for the explanations to provide top-down biasing to the low-level features. He generalized the framework to include multi-level hierarchies of features, where the features at one level are the explanations for the previous level. However, in his implementation, he used only one level of features and one level of explanations.

The framework provided by Olson is quite general, but there were a number of extensions that needed to be made to deal with extra problems induced by spatial indexing and temporal factors. Before describing the extensions, we provide on overview of Olson's feature binding mechanism. The discussion is a general one, but to make the ideas concrete, the reader may think of features and explanations as entities in the shape or motion hierarchies. In these hierarchies, the higher level entities (explanations) are composed of (explain) the lower level entities (features).

### 3.3.1 Olson's Feature Binding Mechanism

Olson introduces his feature binding approach with the following [Olson, 1989]:

> Suppose one is given a set of *features* in an image, and wishes to interpret each of them as a manifestation of some underlying *explanation*. For example, one might observe edge segments and try to explain them in terms of lines... The feature binding idea calls for explanation units to compete for the right to explain (and receive activation from) the feature units. Thus explanation units will compete only when their feature sets overlap.

The idea is to divide up the activation of a feature amongst its potential explanations (owners, higher level features, etc.), in such a manner that more active explanations get more of the activation, but the total activation accorded to all explanation sums to the

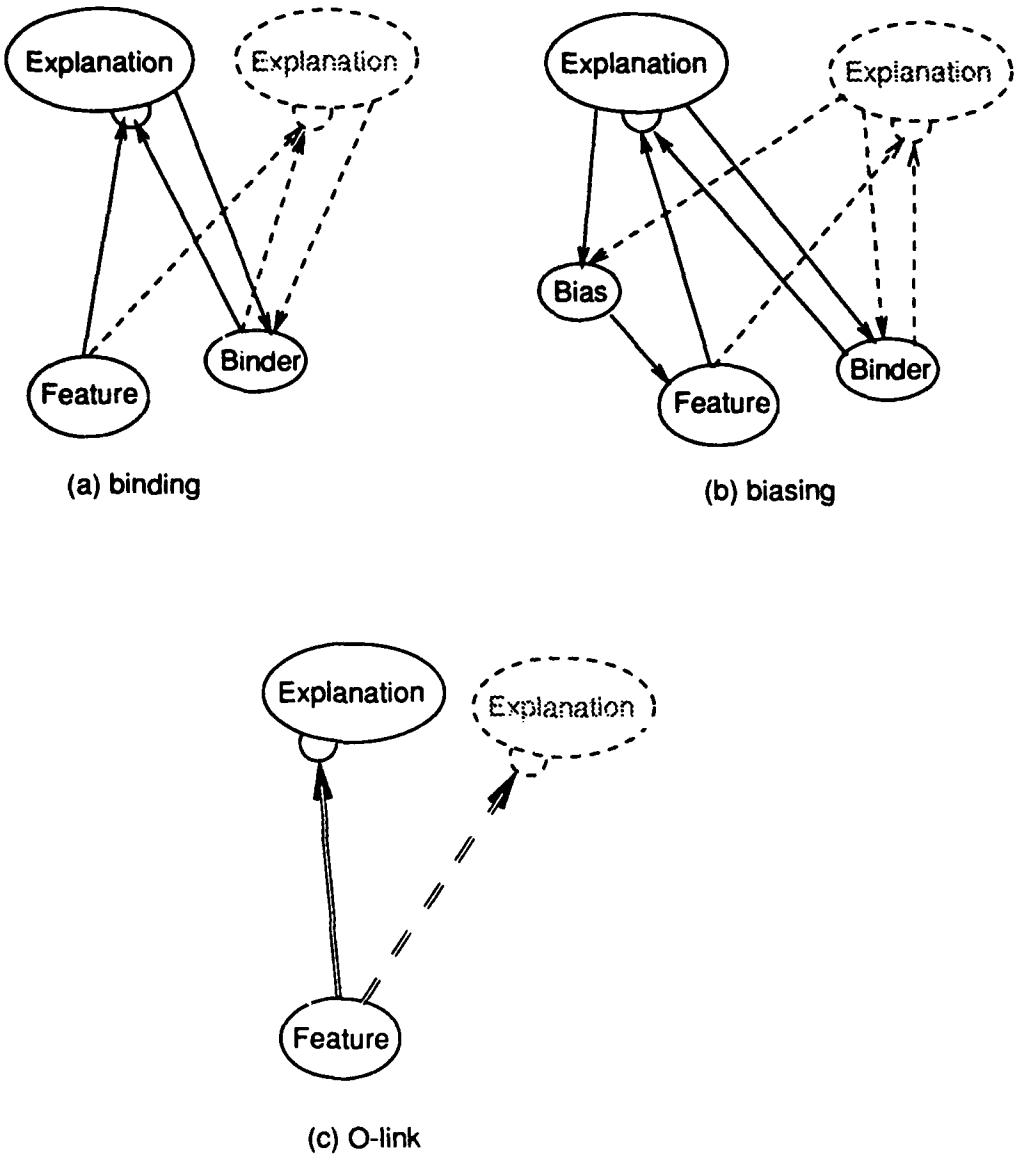


Figure 3.21: Binding Features to Explanations

activation of the feature. The mechanism is very simple. Associate a binder unit with each feature unit. The binder unit computes the sum of the activations of all the competing explanations. With this binder activation as input, each explanation can determine its share of the feature's activation by comparing its activation to the binder activation. An explanation that is much more active than any of its competitors will receive the lion's share of the feature's activation, whether or not its current high level of activity is due to input from this particular feature or to input from other features or top-down feedback. Figure 3.21(a) illustrates the process. A single feature unit and its binder unit are shown together with two explanation units (one drawn with dashed lines) and the associated links.

There is a site on each explanation unit (indicated by the small circular attachment) for each feature that contributes to it. The site receives activation from the binder unit as well. The binder unit receives activation back from all the explanation units that compete for the feature unit's activation. The explanation unit compares its own activation to that of the binder unit, and takes its proportional share of the activation arriving from the feature unit[10]. As the relative activations of the binder unit and the explanation unit change, so does the proportion of the feature activation accorded to the explanation. Olson calls this a *dynamic weight.*

We now present the mechanism more formally. Consider a hierarchy of features, such as the shape and motion hierarchies introduced earlier. The features at a particular level act as explanations for the features at the preceding level. We shall call a feature at level $n$ an $n$-feature[11]. $n+1$-features compete for the activation of $n$-features. Denote the activation of any feature $X$ at any level at time $t$ by $F_x(t)$, and the activation of the associated binder unit by $B_x(t)$. Consider an $n$-feature $\gamma$ and its associated binder unit, and an $n+1$-feature $\theta$ which competes for $\gamma$'s activation. Denote the amount of activation accorded to $\theta$ from $\gamma$ at time $t$ by $\theta_\gamma(t)$. Then according to the algorithm described above:

$$\theta_\gamma(t+1) = F_\gamma(t)\frac{F_\theta(t-1)}{B_\gamma(t)} \tag{3.1}$$

The binder unit for $\gamma$ computes the sum of the activations of the explanations (i.e., $n+1$-features) competing to own it:

$$B_\gamma(t) = \sum_{\theta\in\Theta} F_\theta(t-1) \tag{3.2}$$

where $\Theta$ is the set of $n+1$-features that compete for the activation of $\gamma$. Two things follow immediately. First, the sum of the activations accorded to all the $n+1$-features that compete for $\gamma$'s activation is exactly $\gamma$'s activation:

$$\begin{aligned}\sum_{\theta\in\Theta}\theta_\gamma(t+1) &= \frac{F_\gamma(t)}{B_\gamma(t)}\sum_{\theta\in\Theta}F_\theta(t-1)\\ &= \frac{F_\gamma(t)}{B_\gamma(t)}B_\gamma(t)\\ &= F_\gamma(t)\end{aligned}$$

Second, if there are $p$ $n+1$-features that compete for $\gamma$, and they have equal activation, then the amount of $\gamma$'s activation accorded to each one is simply:

[10]Olson assumes activations in the open interval (0,1), so that there is never a divide-by-zero problem in the calculation of the proportion, at least in theory.

[11]$n$ starts at one at the lowest level of the hierarchy and increases with movement up the hierarchy.

$$\frac{F_\gamma(t)}{p}$$

Such symmetry can be broken in two ways. First, we allow weights on the links from an $n$-feature to its potential explanations ($n+1$-features). Different weights will cause different amounts of activation to go to each of the $n+1$-features even if they are equally active to begin with. Second, an $n+1$-feature receives activation bottom-up from several $n$-features, in general, and top-down from its own set of explanations ($n+2$-features). Unless all these sources of activation provide equal amounts to each competing $n+1$-feature, symmetry will be broken.

In a simple input-output network with no feedback and all features contributing to all explanations, this is a winner-takes-all mechanism. However, it is not equivalent to other winner-takes-all setups. If there is feedback from higher levels activated, for example, by other modalities, or if some features do not contribute to some explanations, then it is possible to reach a stable state in which a feature's activation is divided amongst several competing explanations. Instead of winner-takes-all, it implements strongest-takes-most. For competing explanations, it tends to channel bottom-up activation to the more active explanations. If a feature can play different roles in more than one explanation simultaneously, then separate binder units, one for each role, allow separate competitions amongst different sets of explanations for the feature in each of its different roles. There is one serious flaw in this feature-binding scheme, which became apparent during our experiments. If a feature contributes to many explanations, then initially each explanation receives a very small amount of activation from the feature. If higher level networks depend on a minimum level of explanation activity to become active themselves, then these higher level networks cannot become active enough to provide top-down feedback to break the deadlock by reinforcing one or another of the explanations. It is not clear how to overcome this problem.

### Top-down feedback

The mechanism described above enables bottom-up activation to be routed to the most active explanations. We would also like top-down feedback from the active explanation to be routed to its features. This encourages formation of a globally consistent gestalt, and recall that consistency is one of the two defining criteria of the gestalt phenomenon[12]. But we want to avoid a situation where many low-activation explanation units combine to highly activate a feature which is a part of all of them. Olson's solution is to take the activation of the most active explanation for a feature and use that as top-down bias. The claim is that [Olson, 1989]:

[12]This is the "controlled hallucination" referred to previously.

> Early in the relaxation the higher levels of the network will be in a highly ambiguous state, with low levels of activity distributed broadly across the units. Under these conditions the top-down bias units will have little or no effect. Only when the higher levels have enough information to settle on at most a few candidate explanations will they begin to have a significant impact on what happens at lower levels.

Figure 3.21(b) illustrates the idea. The Figure shows a bias unit added to the network. The bias unit receives the same set of inputs as the binder unit, namely inputs from all the explanations that compete for the feature activation. The bias unit computes the maximum of its inputs to determine its own activation. This value is fed back to the feature unit which uses the input as a multiplicative factor[13]. Denoting the activation of the bias unit for feature $\gamma$ at time $t$ by $M\gamma(t)$ (after Olson), equation (3.1) now becomes:

$$\theta_\gamma(t+1) = M_\gamma(t)F_\gamma(t)\frac{F_\theta(t-1)}{B_\gamma(t)} \tag{3.3}$$

To simplify illustrations, we depict the complex of binder unit, biasing unit and the associated links as a simple hollow link as shown in Figure 3.21(c), which we shall call an O-link.

### 3.3.2 Gestalts and Spatial Indexing

Vision is an inherently spatial process. The discussion of gestalt formation in the previous section ignored an important aspect of the vision problem, namely that multiple objects and scenarios may occur simultaneously at *different locations* in the scene. If each object and each scenario is represented at each location, then gestalt formation can proceed as previously described. However, in some cases it is unreasonable to expect that an entity's representation is duplicated many times over the visual field. If the entity is complex - that is to say, a complex composition of its constituent parts - then its network representation will likewise be complex. Multiple copies of a complex representation would require copious use of scarce hardware; and the learning problem is made even harder in that each copy would have to be learned.

To avoid these problems, we use scenario representations that are not spatially indexed - that is to say, there is one representation of each scenario. But the low-level features that activate these representations are spatially indexed. Suppose we use the Olson's gestalt formation mechanism to bind the features to the scenarios. Consider a feature that occurs in two different scenarios, and suppose that both scenarios are occurring in the scene *in different spatial locations*. The two scenarios will compete to own the activation of the

[13]The unit adds 1.0 to the incoming value before performing the multiplication; this results in no effect if there is no top-down bias.

feature at both locations, and one will win the competition; so that *both* instances of the features will send all their activation to the same scenario. This, of course, is an undesirable outcome. We would prefer that the activation from each feature ends up flowing to the correct scenario - that is, the feature at one location be bound to one scenario, and that at the other location to the other scenario.

We can abstract this problem as one of binding spatially indexed simple features to complex central explanations. Our solution to this problem is to add a spatially indexed compact version of the complex explanations. This allows us to extend Olson's gestalt formation mechanism to encompass multiple entities presented simultaneously but distinguished by spatial location. The compact version of a complex explanation is a single *proxy* unit; this, rather than the entire explanation, is replicated at each spatial location. The trick is to arrange for the proxy's activation to signify the degree of confidence that the explanation it represents is occurring at its location. We defer discussion of how the proxy computes this belief level until section 3.3.4; for now, let us assume that such proxies can be implemented. The central explanations then use their proxies at each location to compete for ownership of features at these locations. This allows different explanations to bind the same feature at different locations. We now present the this idea in more detail.

### 3.3.3 Binding Spatially-Indexed Features to Central Explanations

The general problem is how to bind spatially-indexed features to central explanations. If we introduce at each spatial location a proxy for each explanation, then we can use a slightly modified version of the O-link mechanism to solve the problem. The proxy's activation indicates the evidence that the explanation it represents is present at the location it represents. In the section 3.3.4 we discuss various mechanisms for computing the proxy's activation. Figure 3.22 shows the network structure used to impleme. t feature binding with proxies.

In Figure 3.22(a), we show a single (central) explanation, Explanation A. This is depicted as a single entity, but it may be an assembly of units and links, such as a scenario. The details of how the explanation is represented are not important here. What is important is that the explanation can provide evidence to its proxy, Proxy A, as to whether the explanation is present at the proxy's location, as shown by the thick dashed link. We show five locations (the dashed hexagons), with the proxy unit, a feature unit, and the feature's binder unit for one of the locations.

The mechanism is as follows. The essential idea is to use the proxies to provide the reference activations which the explanations need to compute their share of the feature's activation. In the simple O-link, the reference activation was that of the explanation itself; here we use the proxy's activation instead. The binder unit receives activation from the proxies of all the explanations that compete for the activation of the feature. It sums up the activations, so that its activation represents the sum of the activations of the competing

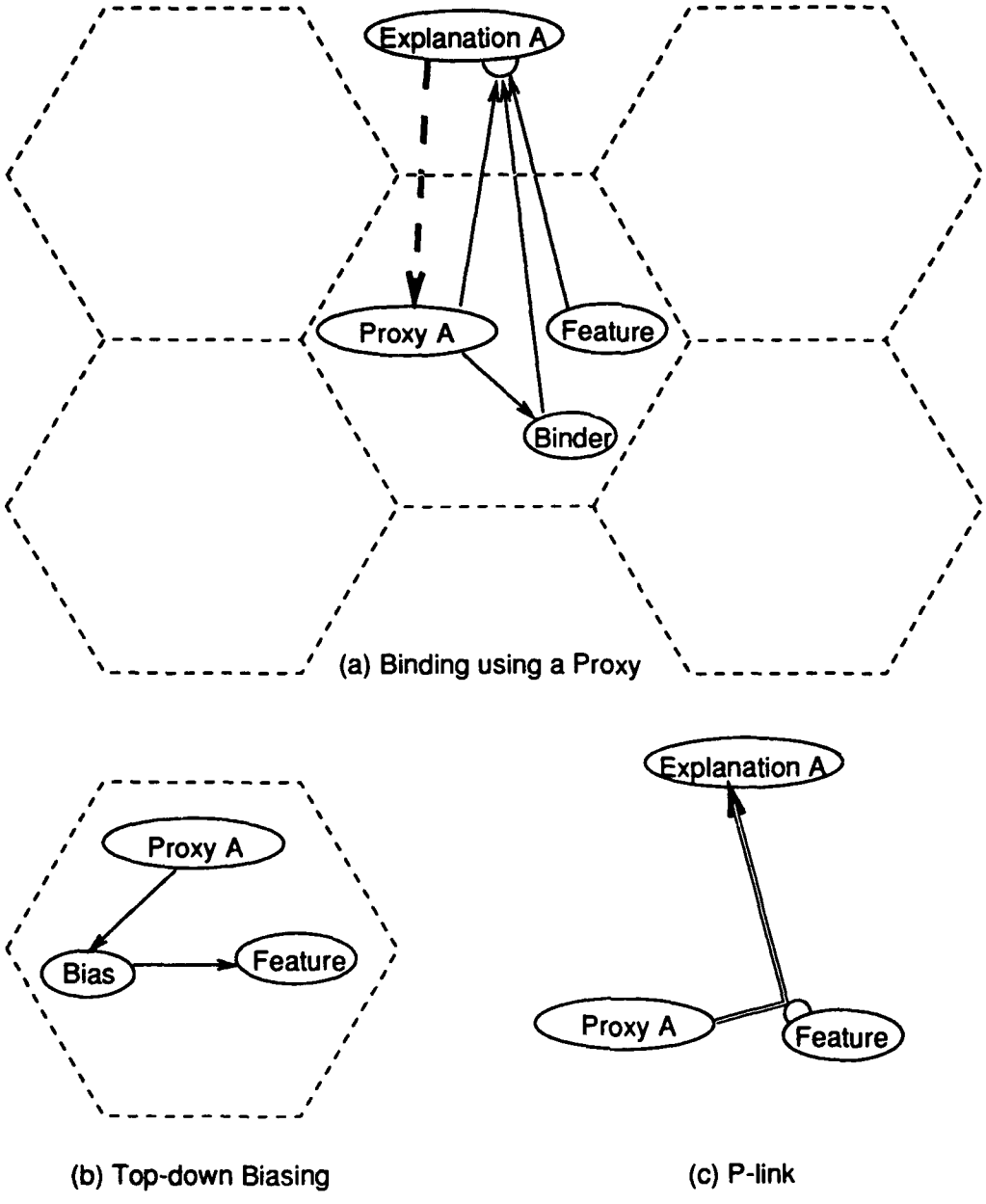


Figure 3.22: Binding Spatially Indexed Features to Central Explanations

proxies. There is one such binder for each feature[14]. The explanation receives activation from its proxy, the feature and the binder. It computes its proportion of the feature's activation as ratio of the activation of its proxy to the sum of the activations of all the competing proxies, given by the binder. In Figure 3.22(a), the three links arrive at a site on

[14]In our application, these binders are separate from the binders used to propagate activity up the shape and motion hierarchies. This is because the role that a feature plays in contributing to a scenario is different from the role it plays in contributing to a higher level feature.

the explanation unit. There is one of these sites for each spatial location, with an identical set of links arriving at each site from the location it represents.

Figure 3.22(b) shows the additional unit and links that are used to provide feedback biasing, similar to the feedback biasing provided by the O-link structure (shown in Figure 3.21). Again, the reference activations used to compute the bias are given by the proxies. The bias unit simply takes the maximum of all the proxies that compete for the feature. The resulting activation is fed to the feature unit as a multiplicative bias.

Figure 3.22(c) shows the graphical notation we use to denote a network structure which combines (a) and (b). We call this kind of structure a P-link, in order to relate it to and distinguish it from the O-link structure.

Some adjustment to equations 3.1 and 3.2 are necessary. We denote the activation of the proxy for explanation $\theta$ at location $L$ at time $t$ by $X_{\theta,l}(t)$. The adjustments consist of substituting the activation of the proxy for the activation of the explanation. The new version of equation 3.1 is:

$$\theta_{\gamma,l}(t+1) = F_{\gamma,l}(t)\frac{X_{\theta,l}(t-1)}{B_{\gamma,l}(t)} \tag{3.4}$$

where the subscripts $l$ refer to the location $L$. The new version of equation 3.2 is:

$$B_{\gamma,l}(t) = \sum_{\theta\in\Theta} X_{\theta,l}(t-1) \tag{3.5}$$

where $\Theta$ is the set of explanations that compete for the activation of feature $\gamma$ at location $L$. It is plain that the total activation provided to all explanations by feature $\gamma$ at location $L$ is $F_{\gamma,l}$.

When the central explanation is a scenario, we must consider potential interactions between the feature integration time embodied in the scenario and the settling time for the gestalt process. For MLD recognition, the gestalt settling time should be much faster than the rate of change of features in the input, so we assume that the two processes do not interfere with each other.

### 3.3.4 Proxies for Scenario Indexing

We previously skipped over the mechanism for implementing proxies. Proxy units can be viewed in different ways. Gestalt formation uses a proxy to direct activation to or away from its progenitor. Attention mechanisms could use the proxies as a representation of what object or action is located where; relative activation of the proxies could serve as the cue for a sequential attention mechanism. In all cases, the information that the proxy maintains should be the degree of belief that its parent is occurring at the proxy's location.

In essence a proxy must compute a reduced version of the match that its parent computes. Were the proxy's version not reduced, the proxy would need to duplicate the representational

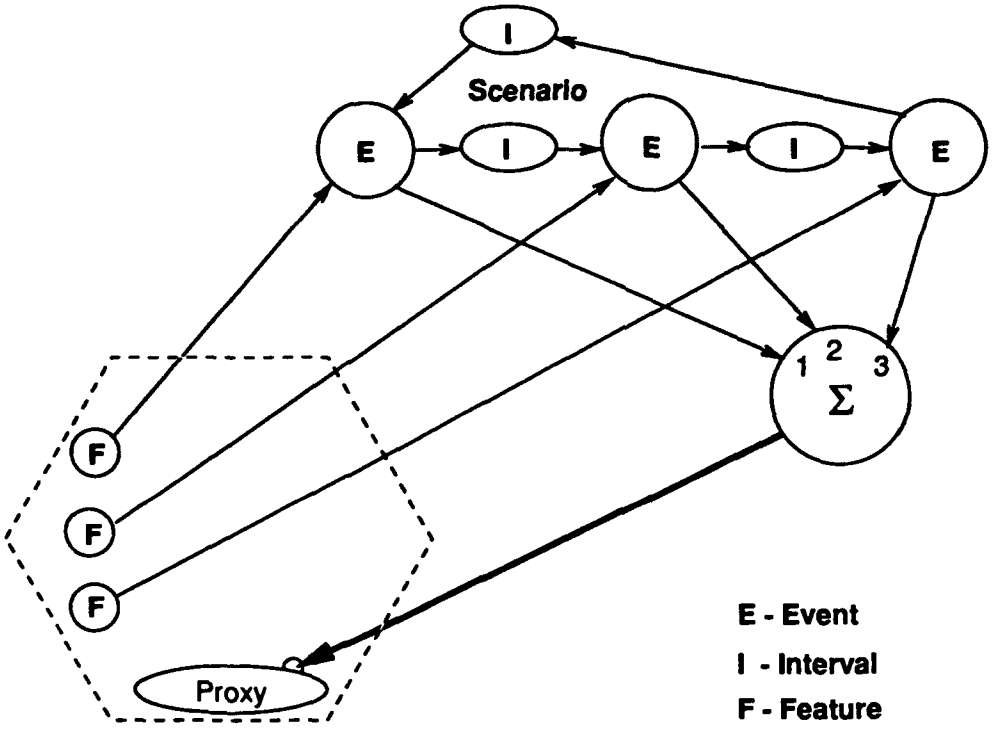


Figure 3.23: Simple Dependence of Proxy on Parent

and processing power of the parent, obviating the need for the latter. However, the proxy may be able to use information generated during the parent's match processing. Proxies that use information from the parent (as shown by the dashed link in Figure 3.22) are said to be *dependent*, while those that do not use information from the parent are said to be *independent*. One class of independent proxies are the summation units introduced in section 3.1.1. The simple summation units compute whether or not the features composing the sequence are present (over time). The ordered summation units also check the constraints on the order in which the features occur. Use of these type of units as proxies has the advantage of needing no feedback links from the parent scenario. A summation unit contains a subset of the information represented in the scenario - the set of events comprising the scenario and in the case of ordered summation units, the ordering imposed on those events. Therefore it can compute locally a degree of belief that the input data at the location correspond to the parent scenario.

The disadvantage of independent proxies lies in their very independence. Their activation is not directly linked to that of their parents in any way. If the gestalt formation process results in a global interpretation that is at odds with the local evidence at a particular location, then an independent proxy at that location may be active even though its parent is not. This may interfere with the correct functioning of those mechanisms that depend on the proxies: attention, and gestalt formation itself.

We can provide a means for the parent to influence the activation of the proxy. Such

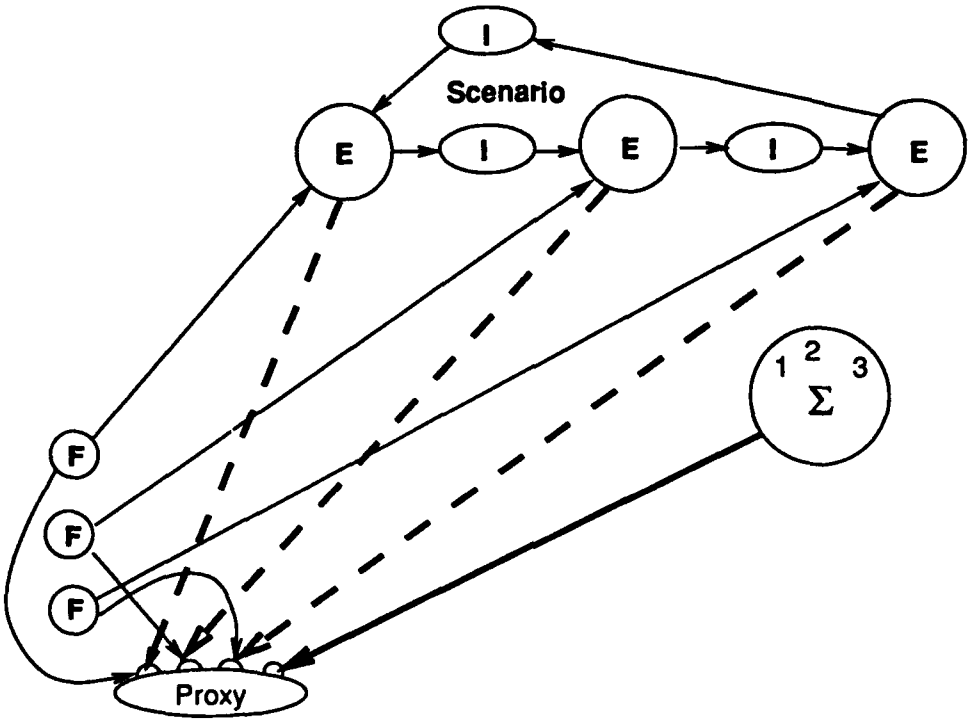


Figure 3.24: Phase Dependence of Proxy on Parent

proxies are said to be *dependent*. The simplest type of proxy dependence uses the overall activation of the parent to modulate the proxy's own activation. If a parent is inactive, none of its proxies will be able to become highly active, whatever the local evidence. For this type of proxy dependence, the dashed link shown in Figure 3.22 carries the activation of the parent, which activation is multiplied, or otherwise factored, into the level of activity determined by the (ordered) summation process. The overall activation of the parent is represented by the parent's evaluator unit. Figure 3.23 illustrates a three event scenario and the feedback (thick link) provided from the evaluator to a proxy unit at a particular location represented by the hexagonal dashed box.

A more sophisticated type of proxy dependence is to use the time information represented in the parent to influence the proxy. If the parent is a scenario, we may consider it as a system that attempts to establish resonance with what is occurring in the input. Simple dependence as described above uses the amplitude of that resonance to modulate proxy activity. Phase dependence uses phase information as well. The proxy compares the currently active features at its location, with the currently active event in the parent. A match is found if and only if the active features at the proxy's location are those that correspond to the active event unit in the scenario. Phase dependence checks the temporal correlation of the input data at the proxy's location with the resonating parent.

Figure 3.24 illustrates the additional links required for phase dependence. The Figure is a development of Figure 3.23, omitting some of the detail to avoid extreme clutter. For

each event represented in the scenario, there is a site on the proxy. This site receives links from the parent's event unit (shown as thick dashed links), as well as the feature unit at the proxy's location that corresponds to that event. Each site compares the change in activation from the event unit with the change in activation from the feature unit. If these two sources change in the same fashion at the same time[15], then the site has positive value. If a change occurs in one source but not the other, or if they change in opposite directions, then the site has negative value. If no change occurs, the site has zero value. The proxy unit then simply sums up the site values, and modulates that sum by the activity of the parent's evaluator unit, to produce its output. It should be apparent that the proxy unit will become active to the extent that the changes occurring in the input are mirrored by, and are contemporaneous with, the changes occurring in the parent, and then only to the extent that the parent as a whole is active.

### 3.3.5 Proxies As An Attentional Mechanism

Although we have motivated the use of proxies by referring to the problem of forming a perceptual gestalt between spatially indexed and central representations, they turn out to have much broader use. Since a proxy's activation is a measure of the degree of belief that a particular central entity (e.g., a scenario) is to be found at a particular location, the proxies can be used as the basis for a selective attention mechanism. Apart from their role in binding particular features to particular explanation, they can be used in two other ways. First, a proxy can be used to modulate the significance of particular features at its location (those features corresponding to its explanation). Second, the proxy can be used to direct feedback to the same features. These two roles are the bottom-up and top-down aspects of selective attention. Enhancement of significance of particular locations to particular central entities causes those entities to respond preferentially to that location, and causes those entities to respond more strongly than entities whose features are not enhanced. Thus the attention mechanism simultaneously selects a location and a central entity for preferential bottom-up processing. Similarly, the proxy can direct feedback from the central entity to the features which compose it at the proxy's location, in effect increasing the expectation of those features at that location. This is selection of particular features at a particular location for top-down processing.

It is important to realize that this is a fundamentally different use of the proxies than in binding for gestalt formation. In the gestalt formation case, the proxy is used to implement competition amongst central entities for a particular set of features at a particular location. The spatial indexing of proxies allows a separate competition at each location. In contrast, when a proxy is used to enhance the significance to a central entity of features at particular location, the effect is to focus the attention *of that entity* on that location, to the (partial) exclusion of other locations. The gestalt mechanism allows multiple entities in the scene

[15]For simplicity we ignore the small time-delay that occurs between the activation of the feature unit and the response of the event unit.

to be activated in parallel; the attention mechanism allows each entity to focus on the most promising locations, ignoring clutter at different locations. The two would be equivalent if every feature detected in the scene could be accounted for by some central entity; but this is rarely the case, even in an uncluttered scene, due to chance configurations of the features in the scene. It is also important to note that the coding of proxies for both location and central entity allows multiple entities to attend to different spatial locations in parallel.

A further refinement is to allow the a proxy to inhibit as well as enhance its features. This can be implemented by endowing the proxy with a "resting" level of activation, and adding competition amongst the incompatible proxies at each location. The more active proxy will drive the others below their resting potential. The inhibited proxy's features at that location then become less significant to the proxy's central entity, and the top-down feedback to the features becomes inhibitory rather than excitatory. The effect is that the winning central entity inhibits all other entities from receiving activation from that location.

# 4 Network Implementation

In this chapter we elaborate each component of the architecture presented in Chapter 2. Figure 4.1 shows once again this architecture. We describe the network structures used to implement the feature and scenario hierarchies and the what/where attention map, making extensive use of the mechanisms developed in Chapter 3. The experiments reported in the next two chapters use these structures.

## 4.1 Feature Hierarchies

The lowest level of the model that is implemented is the Segment level (see Figure 4.1). This level represents single line segments such as those forming the limbs in a stick figure. The middle (component) level represents combinations of line segments, e.g., the left leg of a stick figure. The highest (assembly) level represents combinations of components, for example, the pair-of-legs of a stick figure.

The units representing these features are arranged in a topographic map, using a hexagon-based tiling with increasing tile size and decreasing spatial resolution as one moves up the levels of the feature hierarchies. In the next two subsections, we describe the representation of spatial location and the way in which spatial composition is achieved.

### 4.1.1 Spatial Location

As Figure 4.1 shows, the Shape and Motion feature hierarchies are arranged so that the spatial extent of a feature increases as one moves up the hierarchy; at the same time, the spatial resolution decreases. In Figure 4.2 we illustrate the tiling of the visual plane at the lowest level at which segments are represented. At each level of the hierarchy, a feature is considered to have a spatial extent and a spatial location. The spatial extent is encoded implicitly, while the spatial location of a feature is one of the cells of the lattice. Note that the feature may extend outside the cell that locates it (e.g., a leg component extends well outside the cell containing the hip which is used to locate the component). For a segment, its cell locates the end of the segment about which the segment is rotating.

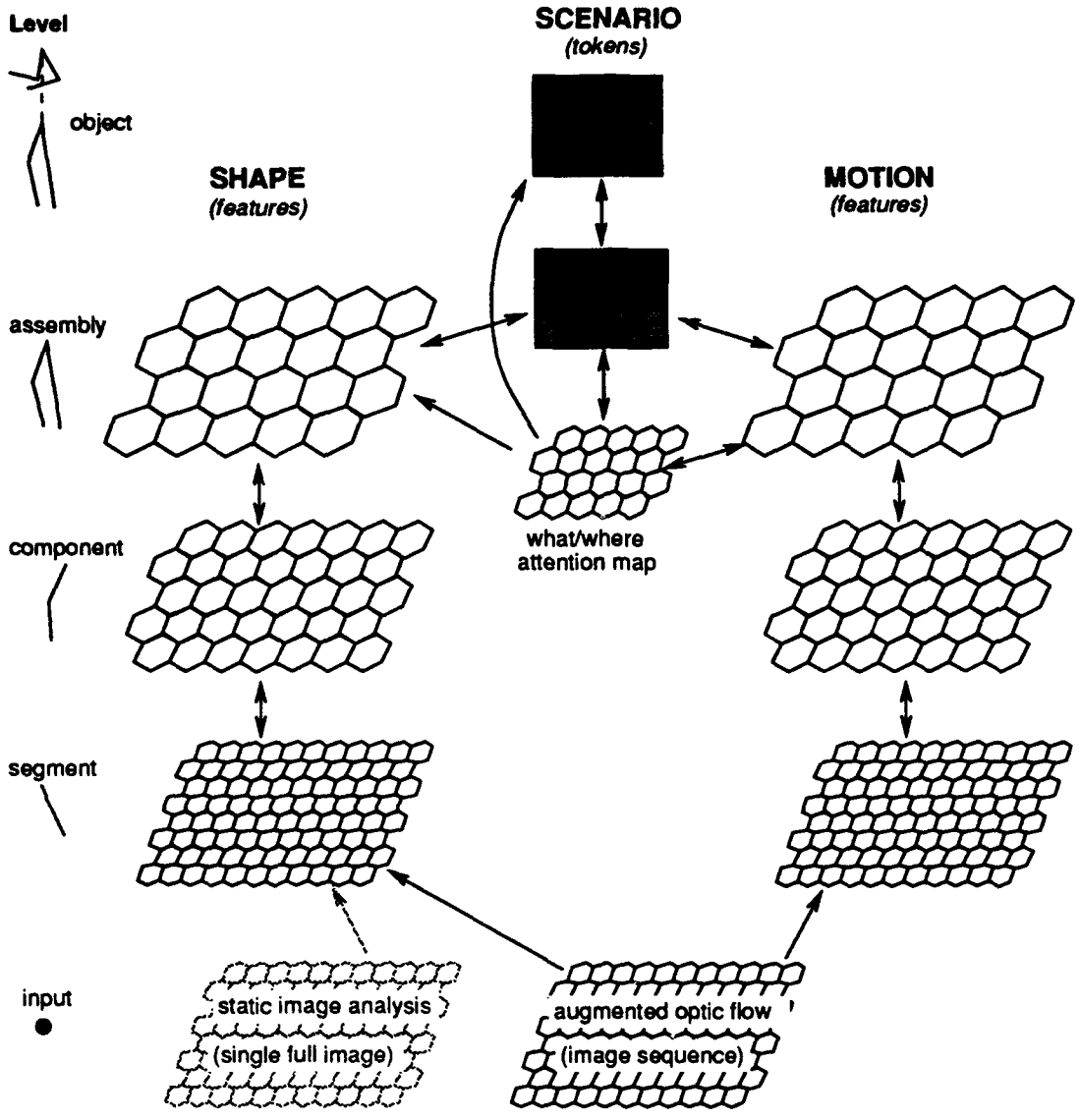


Figure 4.1: Architecture Overview

The next three levels - components, assemblies and objects - are illustrated in Figure 4.3. Although the feature hierarchies do not extend beyond the assembly level, the scenario hierarchy extends one level further, to the object level. As we shall discuss at the end of secion 4.3.5, there may need to be a spatial lattice associated with this level, even though scenarios are not spatially indexed. In Figure 4.3, we see, for the component and assembly levels: the area of spatial localization, at left; the manner in which the cells at the previous level are composed into a cell at this level, in the middle; and the tiling and overlap of the cells at this level, at right. The object level diagrams show the composition and tiling only; cell size can be inferred. It is important to note that spatial localization is *not* the same as

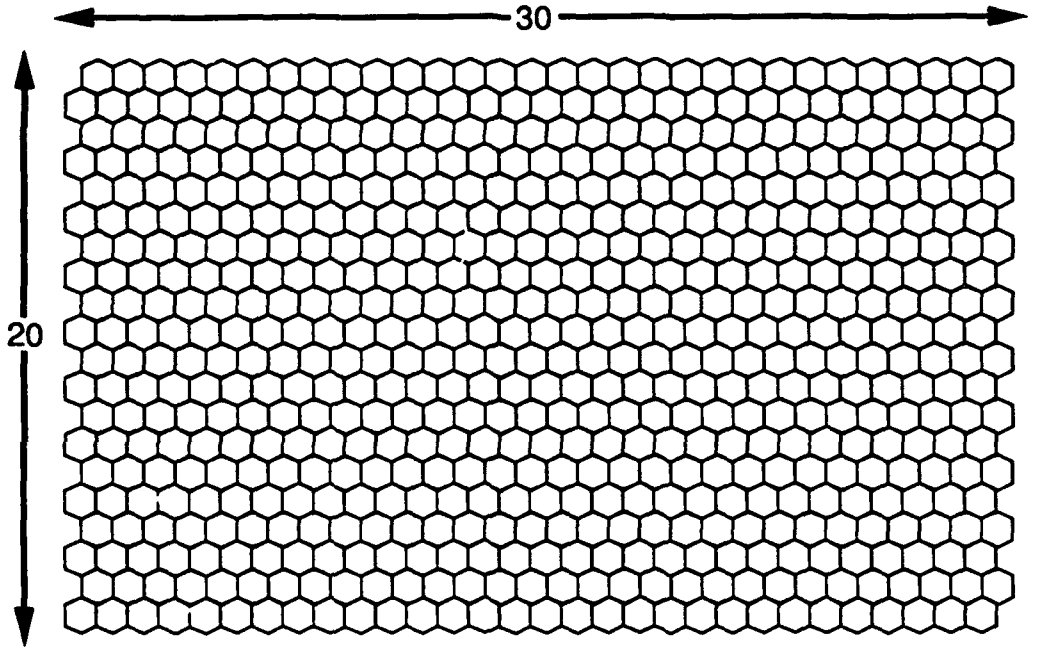


Figure 4.2: Tiling used at Segment level

receptive field. Spatial localization refers to the quantization of space in the representation of where a feature is in the visual field. Receptive field refers to the area of the visual field in which an appropriate stimulus will activate a particular feature unit. The difference is illustrated with the following example: we may represent the location of a leg-like feature to within, say, $\frac{1}{2}^{\circ}$, while the receptive field size for a leg-like feature might encompass several degrees of visual field.

Each level *n* feature is composed of seven level *n-1* features, and cell area increases by a factor of approximately four[1] at each upward step in the hierarchy. Correspondingly, the number of distinct cells at each level decreases dramatically, even though there is overlap of the cells at levels above the segment level. There are approximately 600 cells at the segment level, 150 at the component level, 54 at the assembly level, and 12 at the object level. The rationale for having overlapping cells is to allow for minor imperfections in spatial composition and at the same time not be subject to edge effects. For example, if we a pair-of-legs feature consists of two leg features occurring at the same location (the hip), we wish to allow for some error and accept two legs appearing at neighboring component-level locations. Without overlapping cells, one leg could occur in one assembly-level cell and the other in a neighboring cell, thus activating neither to any great extent, even though the two legs appeared in neighboring cells at the component level.

[1]The actual ratios are 1:7:31:115.

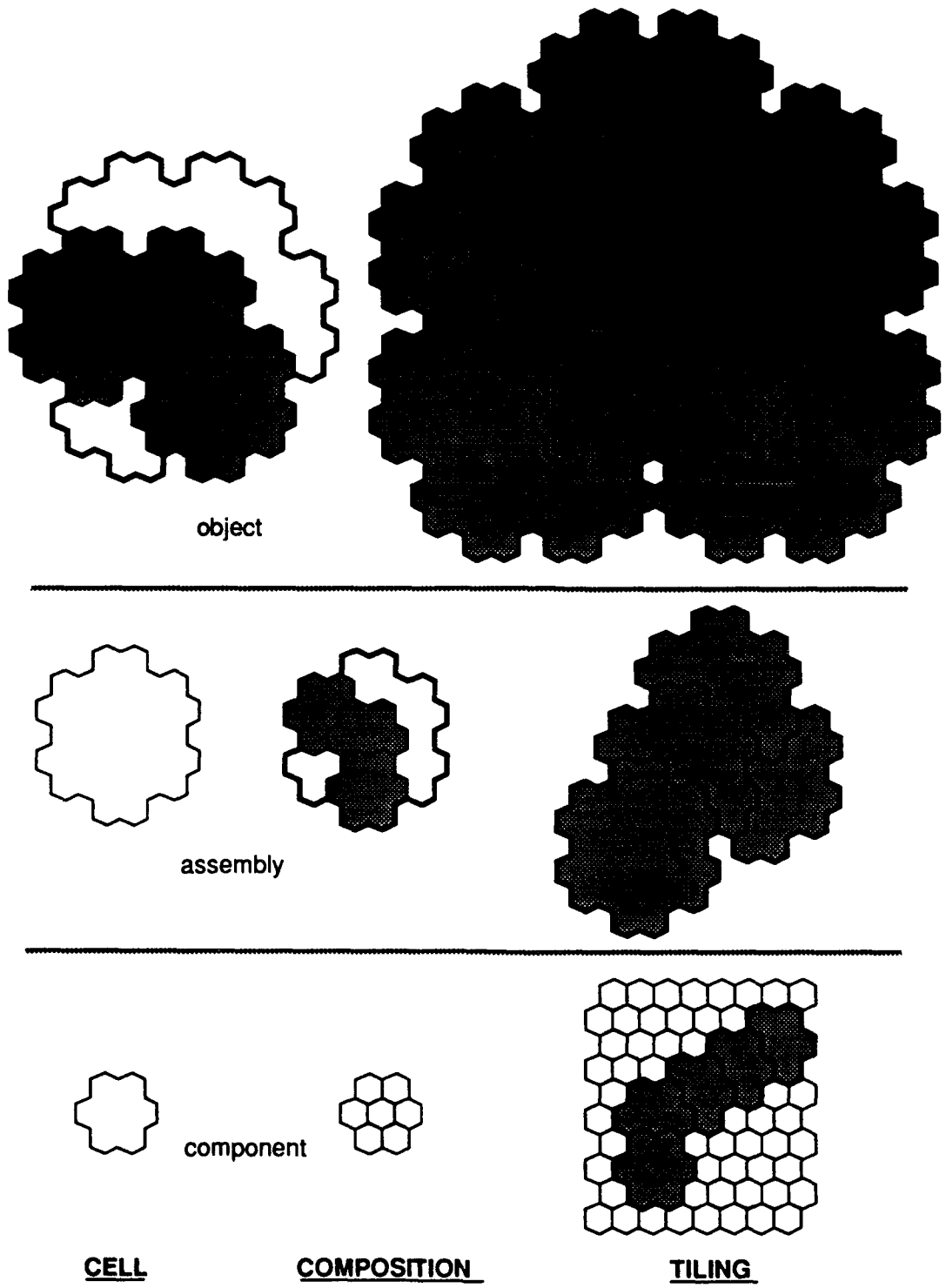


Figure 4.3: Component, Assembly and Object Tilings

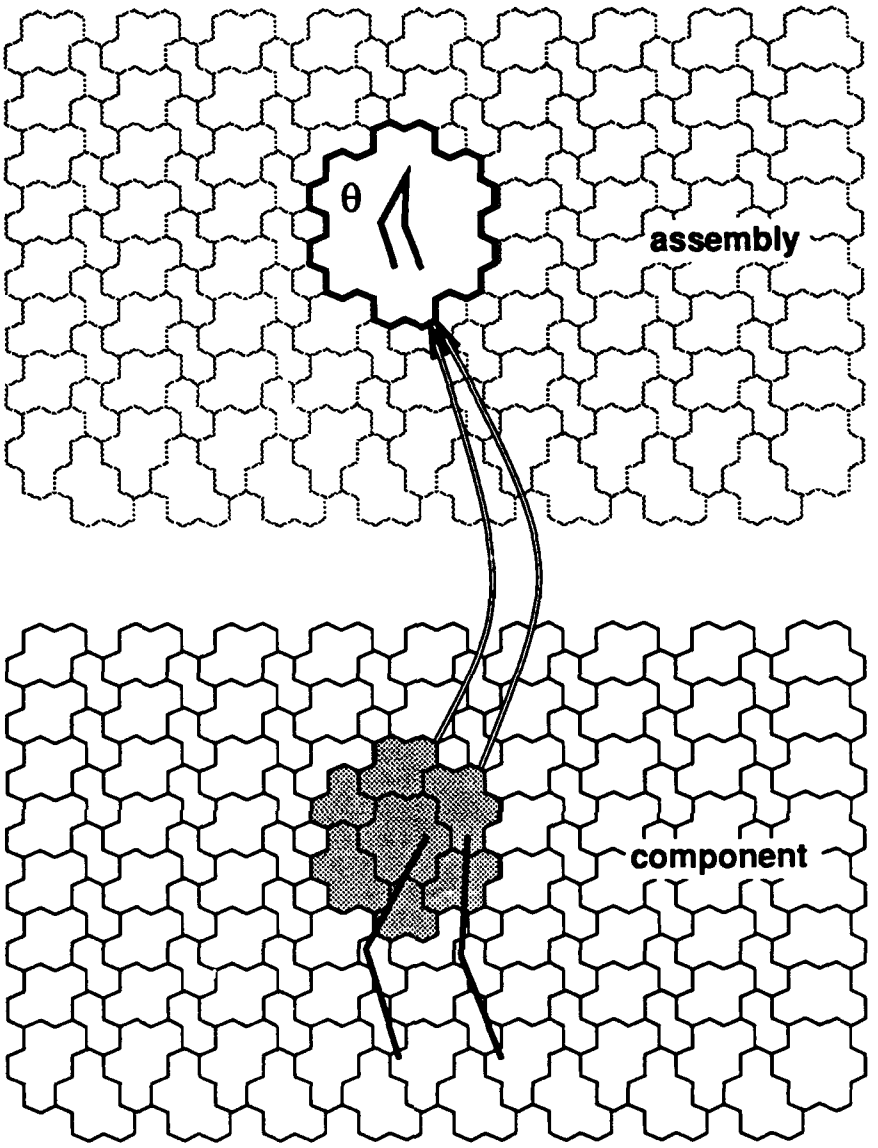


Figure 4.4: Spatial Composition of Features

### 4.1.2 Spatial Composition

In the feature hierarchies, an *n+1*-feature (i.e., a feature at level *n+1*) is composed of a number of *n*-features. The locations of the *n*-features must conform to a set of spatial relationships, and the composed object must be located within the cell in which the *n+1*-feature is located. The spatial constraints are enforced by the explicit spatial indexing of the *n*-features; only those in appropriate spatial locations provide activation to the *n+1*-feature.

An example is shown in Figure 4.4. The lower half of the figure illustrates the component level of the Shape hierarchy, which represents shapes such as single arms and legs. The

upper half illustrates the connections into the assembly level. At this level, pairs of arms and legs are represented. The pair-of-legs unit $\theta$, shown in the upper half, receives pairs of input links, there being one pair for each set of leg component units that are valid in terms of spatial relationship (i.e., at the same location or at horizontally adjacent locations) and in terms of location (i.e., whose cells appear within the area represented by the pair-of-legs unit). One such pair is illustrated.

Composition is enforced by applying a soft AND function (e.g., multiply) to the activations from each of the *n*-features. First we develop the theory of how activations are combined, and then the approximation used in the networks.

### Theory

For a particular *n+1*-feature $\theta$, the cross product of the spatial relationships that define the structure of $\theta$, and the set of *n*-locations that make up the cell in which $\theta$ is located, form a set of sets of (*n*-feature, *n*-location) pairs, denoted by $\Psi_\theta$. Each pair denotes a particular feature unit at a particular location. If every member of any single set in $\Psi_\theta$ is active, i.e., all of $\theta$'s constituents are active, and all the required spatial relationships hold, then $\theta$ should become active. If several sets are active, then $\theta$ should become as active as the most active set (i.e, take the maximum). The activation of *n*-feature $\gamma$ at the location required by set $\psi \in \Psi_\theta$ is denoted by $F_{\gamma,\psi}$. If $\theta$ is composed of *n*-features $\alpha, \beta, \ldots, \mu$, then the contribution of *n*-feature $\gamma \in (\alpha, \beta, \ldots, \mu)$ at the location required by set $\psi \in \Psi_\theta$ to the unit representing *n+1*-feature $\theta$ is:

$$w_{\gamma,\theta} F_{\gamma,\psi}(t)$$

where $w_{\gamma,\theta}$ is the *a-priori* belief that the existence of feature $\gamma$ is evidence for the existence of feature $\theta$.

To enforce composition of the *n*-features that make up $\theta$, we multiply together the individual contributions from the locations defined by $\psi$, and rescale by taking the *p*th root where *p* is the number of *n*-features that make up $\theta$:

$$\sqrt[p]{\prod_{\gamma=\alpha}^{\gamma=\mu} (w_{\gamma,\theta} F_{\gamma,\psi}(t))}$$

To calculate the activation from all the possible location configurations represented in $\Psi$, we take the maximum over the contributions for each of the $\psi \in \Psi$:

$$\sqrt[p]{\max_{\psi \in \Psi_\theta} \left[ \prod_{\gamma=\alpha}^{\gamma=\mu} (w_{\gamma,\theta} F_{\gamma,\psi}(t)) \right]}$$

Note that the weight product may be factored out of the *max* computation, since the *a-priori* belief that the *n*-feature $\gamma$ is evidence for the *n+1*-feature $\theta$ is independent of location:

$$F_\theta(t+1) = W_\theta \sqrt[p]{\max_{\psi \in \Psi_\theta} \left[ \prod_{\gamma=\alpha}^{\gamma=\mu} F_{\gamma,\psi}(t) \right]}$$

where

$$W_\theta = \sqrt[p]{\prod_{\gamma=\alpha}^{\gamma=\mu} w_{\gamma,\theta}}$$

This function may be decomposed into two functions, which can be implemented as site and unit functions respectively:

$$S_{\theta,\psi}(t+1) = \prod_{\gamma=\alpha}^{\gamma=\mu} F_{\gamma,\psi}(t) \tag{4.1}$$

and

$$F_\theta(t+1) = W_\theta \sqrt[p]{\max_{\psi \in \Psi_\theta} S_{\theta,\psi}(t+1)} \tag{4.2}$$

The practicality of this decomposition is limited by the size of the set $\Psi_\theta$ and $p$ (the number of $n$-features that make up $\theta$). Each member of the set requires its own site, with $p$ links arriving at that site. Many of those links are duplicates of links arriving at sites for a different member of the set. If the set is too large to allow a separate site for each member of the set, then a possible compromise would be the following composition function:

$$F_\theta(t+1) = W_\theta \sqrt[p]{\prod_{\gamma=\alpha}^{\gamma=\mu} \max_{\psi \in \Psi_\theta} [F_{\gamma,\psi}(t)]}$$

which requires one site for each of the $p$ $n$-features, and no duplicate links. The price for this reduced representation is a loss of precision in the enforcement of the spatial structure of $\theta$.

## Practice

In practice, we set all the weights to unity as a neutral choice in the absence of any other information. Also, we can effectively have a separate sites for each $\psi \in \Psi_\theta$ because of the nature of our features. We distinguish between *local* and *remote* constituents of a $n+1$-feature. A local constituent is one which must be located within the receptive field of the $n+1$-feature unit. A remote constituent is located outside this receptive field. For example, a leg-like component feature has two segment constituents, one for the thigh-like segment and one for the calf-like segment. The leg-like component is "located" at the hip, as is the thigh-like constituent. Thus the thigh-like constituent is local while the calf-like constituent is remote. We have two types of composed features:

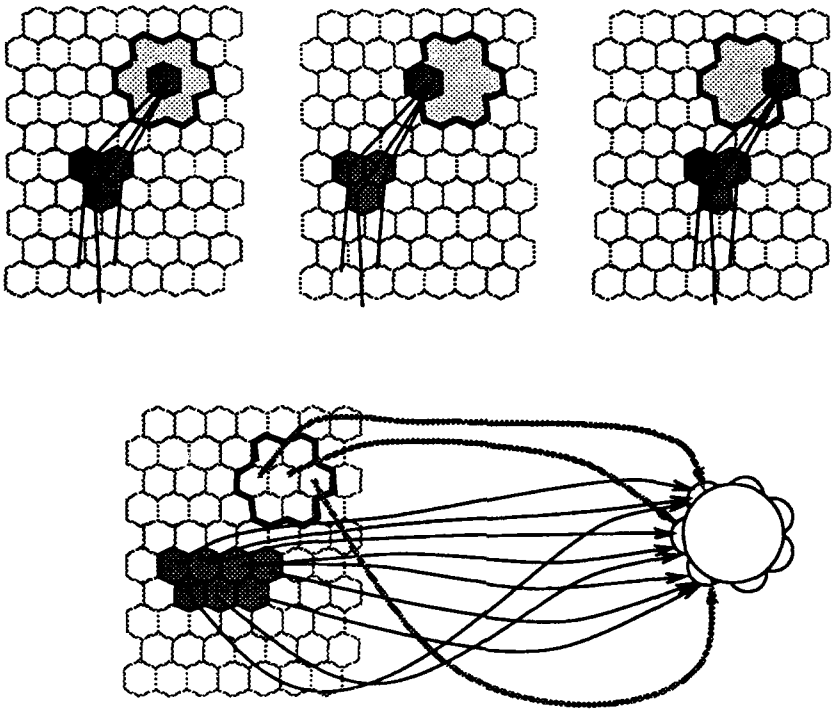

Figure 4.5: Network Structure for Component Features

1. **Components**. All of our component-level features have one local constituent (thigh, upper-arm) and one remote constituent (calf, forearm). The local constituent must occur in one of seven locations (the seven segment-level locations contained in the component-level receptive field). The remote constituent's locations are constrained by the possible locations for the local constituent. Thus $p = 2$ and we need only seven sites, one for each of the possible local constituent locations. At each site, a link arrives from the local constituent at the location represented by the site, and a set of links from the remote constituents (one from the remote constituent at each of its possible locations, given the local constituent location represented by the site). Figure 4.5 illustrates the structure. The top half of the Figure shows three of the seven possible local-constituent locations (the shaded hexagon in the lightly shaded seven-hexagon tile) and the remote-constituent locations consistent with that local-consistent location (the group of three shaded hexagons). The solid lines show examples of the component (a leg-like feature). The bottom half of the Figure shows, on the left, the segment-level locations and on the right, the component-level unit with seven sites, one for each of the local-constituent locations. The lightly shaded links are for the local constituent (one to each site) and the solid thin links are for the remote constituent (three to each site, one from each of the three possible remote-constituent locations). The structure is complex, but all the work is done when the network is constructed. Simulation is quite straightforward. The overall activation

we want is:

$$F_\theta(t+1) = \sqrt{\max_{\psi \in \Psi_\theta} [F_{local,\psi}(t) F_{remote,\psi}(t)]}$$

But there are only seven different locations for the local constituent, so we can regroup the terms around these seven locations and rewrite the activation function, with $i$ indexing the seven different locations, as:

$$F_\theta(t+1) = \sqrt{\max_{i=1}^{i=7} \left[ F_{local,\psi^i}(t) \max_{\psi \in \Psi_\theta^i} \{F_{remote,\psi}(t)\} \right]}$$

where $\Psi_\theta^i$ is the subset of $\Psi$ in which, for each $\psi \in \Psi_\theta^i$, the local feature is at location $i$, and $\psi^i \in \Psi_\theta^i$. This function can now be divided into a site function (seven sites):

$$S_{\theta,i}(t+1) = F_{local,\psi^i}(t) \max_{\psi \in \Psi_\theta^i} [F_{remote,\psi}(t)]$$

and a unit function:

$$F_\theta(t+1) = \sqrt{\max_{i=1}^{i=7} [S_{\theta,i}(t+1)]}$$

There is one further complication. Because of the overlapping tiling used at the component level, a segment level feature provides activation to either one or three component-level locations. If the segment-level source unit is at the same location as the component-level destination unit, then there is only one link. However, if the source unit is at a different location, there are links to three different destination units, so the activation is shared amongst the three destination units. This is implemented in the destination site by dividing the incoming activation by three. A source unit that represents a remote constituent is considered to be located at the position of the local constituent with which it is being combined, for the purposes of this tiling adjustment.

2. **Assemblies**. All of the assembly-level features have two local constituents. For example, two leg-like components are composed to form a pair-of-legs-like assembly, and all three are "located" at the hip. The only complication is the fact that activation may be distributed in the component-level features because of the overlapping tiling scheme. To compute the activation, this distributed activation must be summed over each set of three contiguous component-level locations that are contained within the assembly-level receptive field. Figure 4.6 illustrates the structure. In (a) we show three of the seven component-level locations (shaded) contained within one assembly-level location (outlined in thick lines). These three form one triple, labeled 2. There

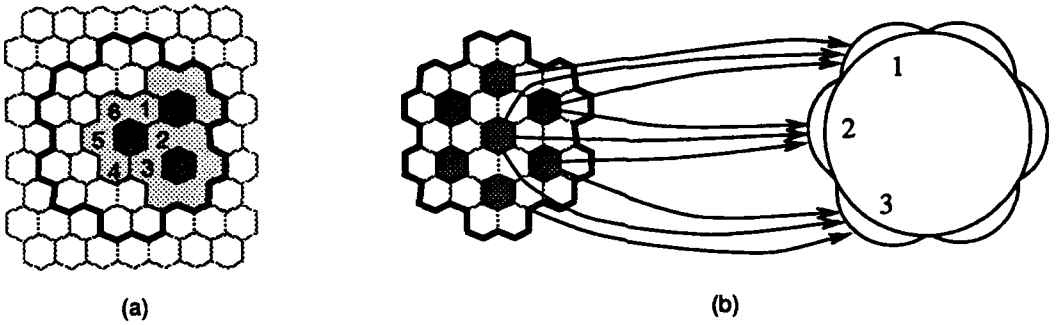


Figure 4.6: Network Structure for Assembly Features

are six such triples. In (b), we show the links for triples 1, 2 and 3. Each site on the assembly-level feature unit represents one triple of contiguous component-level locations. The site takes the sum over the locations in the triple for each constituent, and then combines the resulting activations by taking the square root of the product of the two constituents. This could result in crosstalk errors if there are two similar component-like entities close together in the scene, but it has not been a problem in the simulations.

The implementation of composition with the multiply operator is too harsh. If a constituent is not present (activation zero), then the result of the multiplication is zero, no matter how active the other constituent is. Instead, we provide a 90% penalty, so that the resulting activation of the feature unit is 10% of the activation of the single constituent that is present. This parameter is of prime importance in controlling the trade off between tolerance of missing data and susceptibility to false targets (the pervasive false-positive/false-negative trade off). If it is set high (i.e., a large penalty for missing constituents), then there is little tolerance to missing data, e.g., because of occlusion; if it is set low, then the feature unit will respond to all sorts of stimuli that bear only passing resemblance to the configuration it is designed to detect.

## 4.2 Scenario Hierarchy

The highest level features index, or activate, the lowest level of the scenario hierarchy (see Figure 4.1). Assembly level scenarios represent concepts such as "pair-of-legs-while-walking" and "pair-of-arms-while-skipping". These scenarios in turn activate the higher (Object) level scenarios. In the next three subsections we describe in detail the scenario representation and how it accumulates activation over time, the way in which the lower scenarios are activated by the features, and how the higher level scenarios are activated by those at the lower level.

### 4.2.1 Representation

A scenario is represented using the structure outlined in section 3.1.3. Enabling and support activation are provided by feature units in the Shape and Motion hierarchies, which are spatially indexed (see section 4.2.2). Enabling and support activation originating at a given location arrive at a site on the event unit particular to that location; there are many such sites, one for each location in the Shape and Motion hierarchies, since the scenario represents the entire scene[2].

#### Enabling Conditions: Detecting Change

An enabling condition is the detection of some significant change in the scene. In general, changes can come from many sources - occlusion; motion; shape; color. We restrict changes that act as enabling conditions to changes in motion. In the Motion pathway, all features use rotational motion. The significant changes in motion are changes in the rotational velocity - increasing or decreasing the speed of rotation, or reversing the direction of rotation[3] [Rubin, 1986]. We label the motion before the change the *pre-change* motion and the motion after the change the *post-change* motion. An enabling condition is detected with the the transition of activation in the motion representation from the pre- to the post-change motion. The types of motion changes that are detectable depend upon the representation of motion in the Motion pathway. At the levels that scenarios operate, our representation of motion is by value unit coding [Ballard, 1986], which is to say that each distinguishable motion has a separate unit to represent it. There is no continuum of speed, unlike the lower levels. Different rates of execution of the motion are represented by different units. Positive and negative rotations (i.e. clockwise and anti-clockwise) are represented by separate units. An enabling condition is detected by monitoring the activations in the unit representing the pre-change motion and that representing the post-change motion. Decrease in pre-change activation, increase in post-change activation, or both, are grounds for the enabling condition to be detected. Gradual change in motion is accommodated by having detectors in the motion pathway that are tuned to different angular velocity ranges. As a velocity threshold is crossed, activation passes from one unit to another.

A special sort of change is that from *no-stimulus* to *stimulus-present*. In both the real world and the laboratory setting, the initial appearance of an object (or stimulus) must be considered a high priority event, simply from ecological considerations. The mechanism that detects this kind of change, and assigns it a priority, is outside the scope of this work. In the context of this architecture, the onset of a motion feature because of the commencement of a stimulus does not necessarily indicate that the corresponding enabling condition has just occurred in the input. Therefore, we inhibit detection of enabling conditions for the first few simulation steps of each new presentation.

[2]This is reasonable: at the assembly level, there are only 54 different locations across the visual field.

[3]A rotating wheel, such as on the cart object, will experience no motion events as long as the wheel rotation is of constant angular velocity. A scenario requires change - accelerating from rest, for example.

### 4.2.2 Indexing with Distributed Features

A scenario at level *n* is activated by shape and motion features at level *n*, if there are any. If not, then it is activated by the scenarios at the preceding level. In Figure 2.1, the object-level scenarios are indexed by assembly-level scenarios; the assembly-level scenarios are indexed by assembly-level features. Figure 3.8 shows that scenarios require enabling and support activation. Enabling activation is derived from changes in pattern of activation of the Motion features. Support activation originates in the (instantaneously static) pattern of activation of both the Shape and Motion features.

Enabling activation arrives at the scenario event units via two links. One of the links is from the unit representing the motion expected before the change, and the other from the unit representing the motion expected after the change. The enabling condition is deemed to have occurred if activation on the 'pre-change' link decreases, activation on the 'post-change' link increases, or both of these. The processing is only concerned with the kinematics of this change, and not with the absolute levels of activation of either the pre- or post-change motion features. Support activation arrives from features in both the Shape and Motion pathways.

Links for enabling and support activation arrive from all locations in the scene, that is to say from all locations represented at that level in the Shape and Motion pathways. Each *different* location has its own site on each event unit, as shown in Figure 3.8, so that support activation at a particular location can only be combined with enabling activation at the same location, thus reducing crosstalk. Figure 4.7 shows in more detail the links from two locations in the Shape and Motion pathways to a particular event unit. The small circles represent the Shape and Motion feature units, in locations delineated by the dotted hexagonal grid. The dotted links from the Motion features are from the pre-change motion units; solid links are from the post-change units.

Because of the overlapping tiling used in the feature hierarchies, activation from a single input location will result in activation of one, three or four different assembly-level locations. The three cases are illustrated in Figure 4.8. In (a), we see the case where the input is exactly at the location of an assembly-level unit. In this case, the single assembly-level unit receives all the activation. In (b), we see the case where the input is adjacent to the location of an assembly-level unit. In this case, the activation is divided 5:2:1:1 amongst four assembly-level units. In (c) we see the case where the input is equidistant from the location of three assembly-level units. In this case, the activation is divided 3:3:3 amongst the the three assembly-level units. All other input locations are isomorphic to one of these cases.

This distributed activity is re-integrated by the event unit. If there is just one active input location, re-integration is simply a matter of summing over each diamond-shaped set of four assembly-level locations and taking the maximum value found. This yields the correct result in all three cases. However, if there is more than one active input location (as is the case with MLDs), and the input locations are close enough to activate overlapping diamond-sets of assembly-level units, then summing over each diamond-set may result

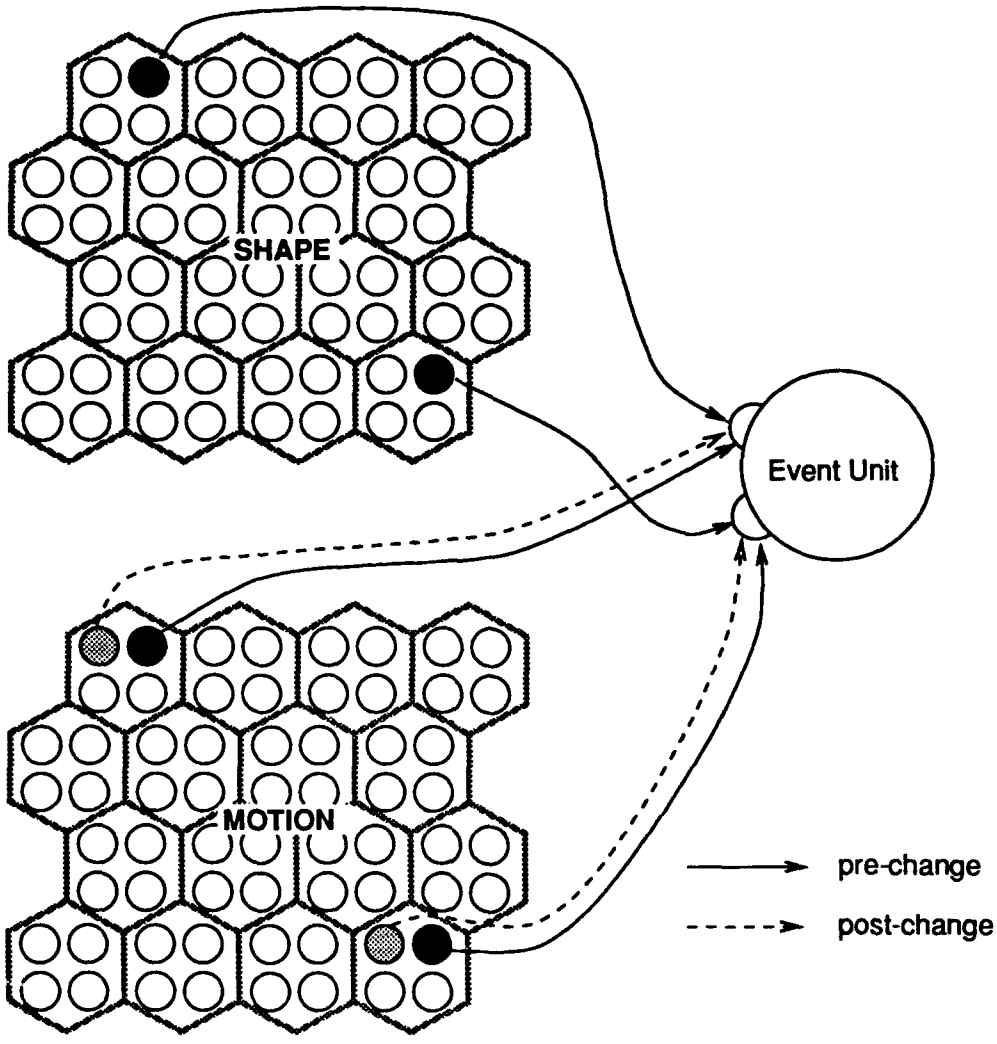


Figure 4.7: Enabling and Support inputs to Event unit

in the activation from the two different input locations being added together. In the first set of experiments, we summed over the diamond-sets. But in the second set, we found there was too much interference between nearby locations. The solution was to sum each *triangle*-shaped set of three assembly-level units. This yields the correct result in cases (a) and (c), and gives 88% of the activation in case (b), a minor inaccuracy. The reduction in crosstalk was considerable.

A problem occurs if the same scenario is occurring at two different places in the scene, but out of phase. The mechanism is suited to parallel indexing of different movements at different locations, but may be unsuited to parallel indexing of the same movement at different locations (unless they are in phase). In the worst case, the indexing mechanism will completely fail if there are enough copies of the same movement occurring in different phases at different locations. We appeal to sequential attentional mechanisms to deal with

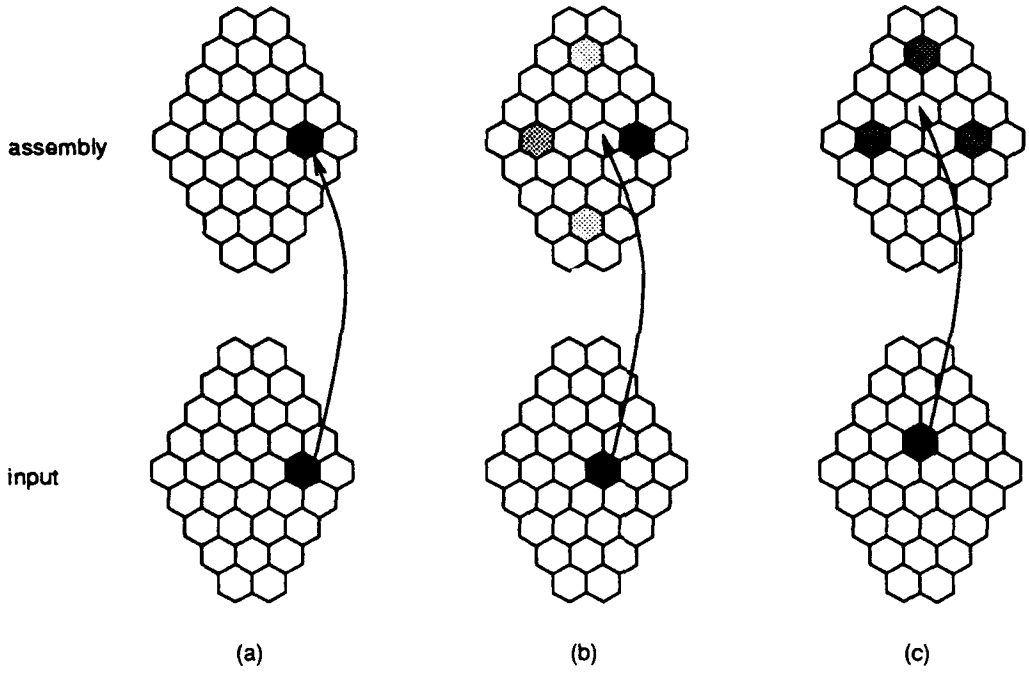


Figure 4.8: Distribution of Activation in the Assembly Level

this problem. The attention mechanism described later in this chapter (see section 4.3) could be used for this purpose, if an appropriate sequential control strategy were developed [Ahmad and Omohundro, 1991].

### 4.2.3 Composition

Higher-level scenarios are composed of lower-level scenarios and are activated by those lower-level scenarios. An object-level scenario, e.g., *biped-walking*, is composed of assembly-level scenarios, in this case *legs-walking* and *arms-walking*[4].

During a particular timespan, e.g., the period of a cyclic movement such as a bipedal gait, object level scenarios have fewer events than their constituent assembly-level scenarios. An object-level event is activated by the assembly-level events that occur between the time of the preceding object-level event and the current object-level event. The object-level event then provides, via interval units, top-down priming to the immediately succeeding assembly-level events. The structure is shown in Figure 4.9. Event units are circular and interval units elliptical. The numbers inside the units represent, for events, the time at which the event takes place, and for intervals, the time interval represented by the unit[5]. The thin solid links are the usual scenario priming links. The thick solid links show the bottom-up activation of the higher level scenario by the lower-level scenario, and the dashed links

[4]That is, the movement the arms make during the walking cycle.

[5]For simplicity, we collapse the time range to a single value in this diagram.

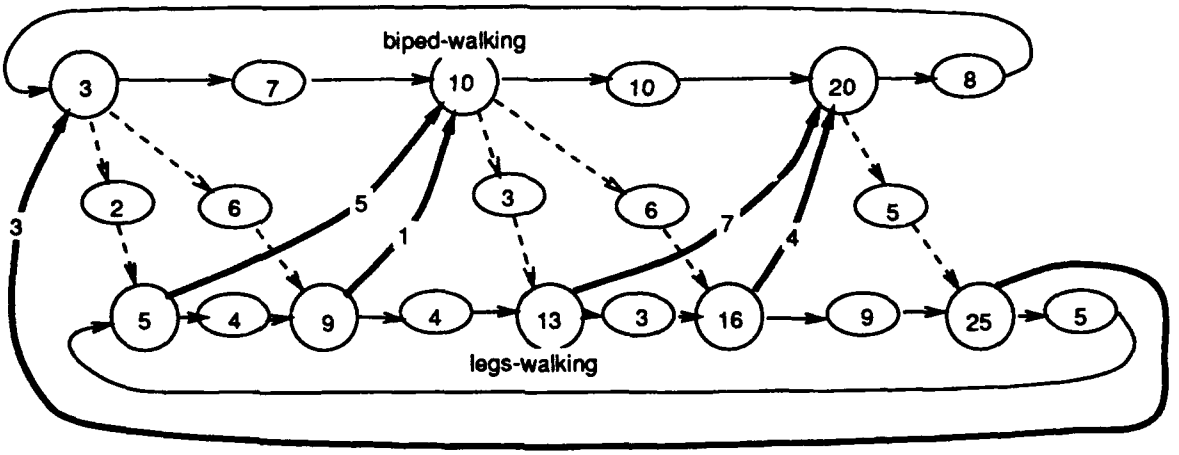


Figure 4.9: Network Structure for Scenario Composition

show the top-down priming in the reverse direction. At the top is an object-level scenario, *biped-walking*. At the bottom is one of the assembly-level scenarios that activate it, *legs-walking*. The other assembly-level scenario, *arms-walking*, is omitted for clarity. The expectancy window for the interval units in the top-down priming path is controlled by the summation unit for the object-level scenario, but these links are not shown. The bottom-up links are implemented as delay lines with a fixed delay, shown by the number breaking the link. Some of them could be implemented with interval units instead, but this was not necessary in the first set of experiments, and in the second set only the event with the lowest uncertainty of onset time was used. It is important to realize that a delay link from at least one assembly-level event is required in order to provide the spike in activation which indicates the object-level event's onset.

This structure composes lower-level scenarios into higher-level scenarios, and, crucially, allows the latter to force synchronization of the former. If the constituent scenarios are out of sync, the high level scenario does not attain full activation. This would occur if, for example, the arms were swinging in-phase with the legs (abnormal) rather than in opposite phase (normal). If one of the assembly-level scenarios is quick to respond to the input (which is the case for the leg-movement scenarios), then it partially activates the object-level scenario, which then encourages the other assembly-level scenarios, via the top-down links, to become active in the right phase. Another positive result of the presence of the top-down links is that if occasional lower-level events are not detected in the input, then the priming path from from the preceding low-level event via a high-level event to the succeeding low-level event compensates partially for the loss of input.

### Spatial Constraints

This described so far structure has one major weakness: it does not enforce spatial constraints on the composition of lower-level scenarios. A pair of arms and a pair of legs,

each executing the appropriate motion cause the *biped-walking* scenario to become active no matter what their spatial relationship. Legs below arms, above, to left or right, the result is the same, so long as the legs and arms are sufficiently spatially separated that they do not interfere. In most of the experiments, we did not enforce spatial constraints in scenario composition, but found them to be necessary in the final experiment using actor-independent scenarios. Our solution uses the proxy units that form the attention map. We defer further discussion to the next section which details the attention map.

## 4.3 What/Where Attention Map (WWAM)

The what/where attention map evolved from early work on using proxy units to bind features to scenarios. It became a general attentional mechanism for focusing bottom-up and top-down processing on particular spatial locations and scenarios. In contrast to most models of attention in which spatial locations compete for processing, the WWAM is designed to allow multiple locations to be attended simultaneously, and the competing entities are high level representations (scenarios) which compete to "own" the activation of features at each spatial location. The result is a parallel mechanism which focuses both bottom-up and top-down processing on spatial locations at which familiar movements are occurring. The mechanism relies on an explicit representation of *what* is happening *where* - hence its name. This representation uses the proxy units, introduced in Chapter 3, to indicate the level of belief that a particular scenario is occurring at a particular location.

### 4.3.1 Proxies as Temporal Correlators

The activation of the proxy unit is the key to the success of the WWAM. A proxy is supposed to represent the accumulated evidence that the parent scenario is occurring at the proxy's location. We use phase dependence, as described in section 3.3.4 to achieve this. The key idea is to measure the correlation, in the time domain, of events in the scenario with the events in the input. The proxy performs this measurement, and sets its activation accordingly. A proxy at a location where there is no input, or at a location where the incoming events do not correspond temporally to the parent scenario, does not become active to any significant extent. But a proxy at a location where the incoming events correspond to the parent scenario, in character and timing, will become highly active.

The network structure used to activate proxies is illustrated in Figure 4.10. At the top of the Figure is a three-event scenario, $S$. At the bottom is the set of motion feature units at location $L$. In the middle is the proxy unit representing the degree of belief that $S$ is occurring at $L$. On the right is the summation unit for the scenario, marked $\Sigma$. The proxy unit has a site for each event in $S$, plus one site at which a link from the summation unit arrives. Each event-site receives a link from the event and from the succeeding interval unit. The activation on these links is used to indicate when the scenario has detected the

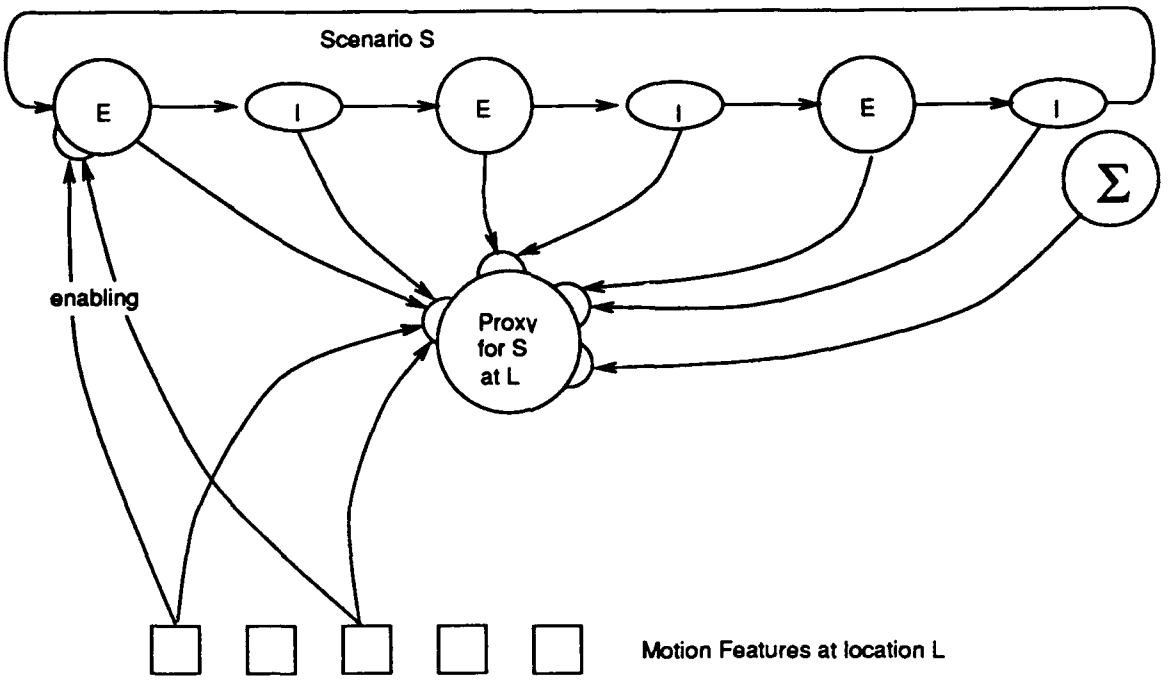


Figure 4.10: Network Structure for Proxy Activation

event (as happens in the summation units). Each event-site also receives input from the motion features at $L$ which are used by the event to determine if an enabling condition has been met. The site compares the activation on the enabling links with the activation on the event and interval links and sets its value accordingly. The site determines if there has been a compatible change along the links[6]: if so, its value is set positive; if not, its value negative, unless there has been no change at all in which case it retains its previous value. The unit then takes the maximum value of the sites and combines it with the summation activation to form its value. The summation activation is used to limit proxy activation: the proxy cannot be only slightly more active than the summation unit (it is logically absurd to have higher belief that something is present at a particular location than that it is present at all). We allow a proxy to be a little more active than its scenario summation unit so that the proxies can begin to work as soon as the scenario responds to the input, rather than waiting until the summation unit has responded to the scenario activity. Like the summation unit, the proxy output value is an exponential approximation to the activation value.

[6]Slightly different site functions were used in the first and second sets of experiments. In the first set, the change in event activation was used to determine if enablement had occured. Since this partially duplicated the processing in the interval unit, in the second set the activation from the interval unit was used to indicate event enablement.

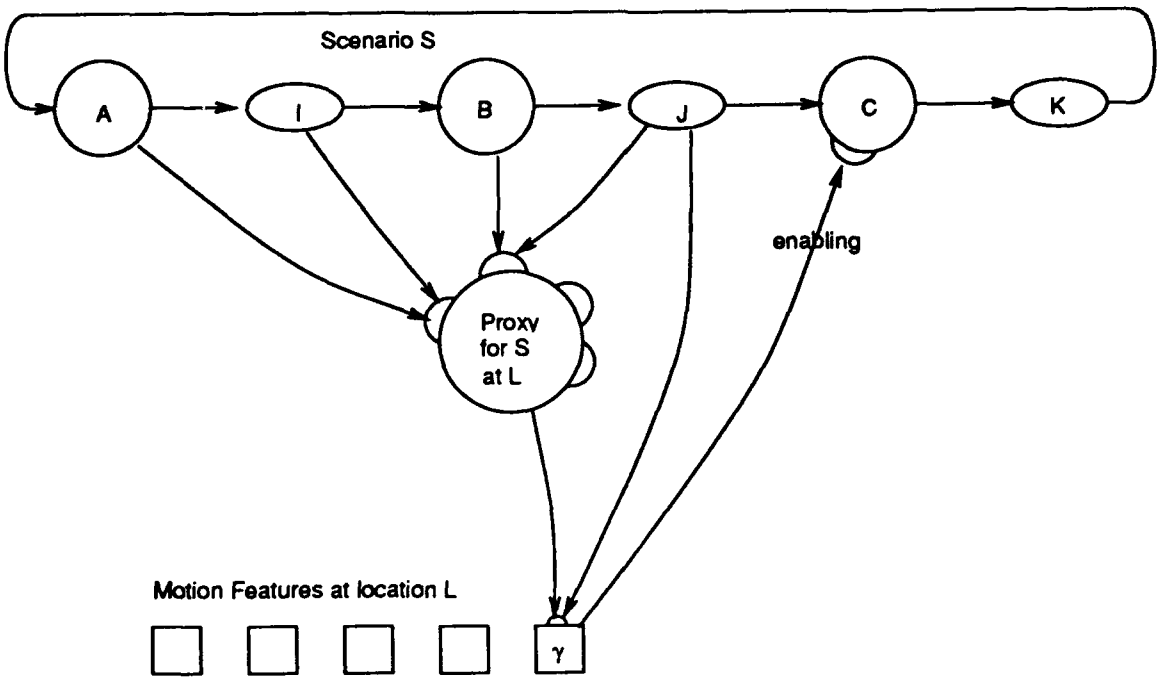


Figure 4.11: Predictive Biasing by Scenario and Proxy

### 4.3.2 Using Proxies to Focus Attention

As has been discussed previously, one of the uses of the proxy units is to focus attention on particular locations and particular scenarios. We can enhance the relevance of the scenarios location by using it to boost the activity of the features at the location. This would be similar to the action of the AM module in [Mozer, 1991]. But because our proxies code not only for location but also for a particular higher-level entity that is being indexed (a scenario), we can enhance the significance of the location selectively for that particular scenario. Boosting feature activation does not achieve that selectivity, since the increased activation is received by all scenarios that use the boosted features. Instead, we add a link from the proxy to the site on each event that codes for the proxy's location. The value arriving along that link is used by the site function to increase the significance of feature activation. The result is that proxy activation increases the relevance of particular features at a particular location to a particular scenario.

### 4.3.3 Using Proxies to Focus Priming

The scenario is able to provide predictive biasing, that is, to facilitate those shape and motion features that are expected to occur in the immediate future. The proxies are used to direct the facilitation to the appropriate spatial locations. Figure 4.11 shows the units and links involved in predictive feedback. Assume that the unit for event *B* has fired in the

recent past, and that the interval unit $J$ is in its expectancy window, i.e. is sending priming activation to event $C$. Assume that the proxy shown in the figure is active. The goal is to facilitate those features that enable and support the next expected event, i.e., event $C$. The Figure shows one such link, an enabling link, from feature $\gamma$ to event $C$. Facilitation of these features is provided by the conjunction of activation from the proxy and from the interval unit $J$. The interval unit input ensures that the facilitation is strongest at the time when event $C$ is expected to occur, and the proxy input ensures that there is no facilitation unless there is evidence that the scenario is occurring at the location. In the Figure we show a site on feature $\gamma$ where facilitatory input arrives. If $F_\gamma(t)$ is the activation of $\gamma$ without this predictive priming, then the new activation is:

$$F'_\gamma(t+1) = \left(1 + \sqrt{\alpha P_{S\perp}(t) I_J(t) + \beta I_J(t)}\right) F_\gamma(t+1)$$

where $\alpha + \beta = 1$. The parameter $\beta$ controls how much priming there is even in the absence of proxy activation. In the first set of experiments, it was set to zero, but in the second set it was set to 0.3 and found to increase the speed with which the correct scenario was indexed.

### 4.3.4 Competitive Proxies

The use of the proxies described above is to *increase* attention by the proxy's scenario to the proxy's location. It is desirable to be able decrease attention to a location if some other scenario provides a superior accounts of its feature activity (under the assumption that there is only one scenario occurring at each location, a reasonable assumption to make). We implement this by endowing the proxy units with a resting level of activation, which can be depressed by competition from other proxies, or increased by resonance between the feature activity and the scenario activity. Now all the units which use the proxy activation subtract the resting level to arrive at the true proxy value. If it is negative, then inhibition occurs (i.e., the features become *less* significant to the scenario events). If it is positive, facilitation occurs. Adding the resting level is simply a biologically inspired way of communicating a negative effect. Computationally it is equivalent to allowing proxy activation to go negative.

In the first set of experiments, proxy competition was not used. In the second set it was used, and the result was a dramatic increase in separation between the winning scenario and the losers.

### 4.3.5 Using Proxies for Spatial Composition

One of the deficiencies of the scenario composition developed above is the absence of any spatial constraints on the composition of higher order scenarios. One solution is to use the proxies in the what/where map to limit activation of the composed scenario unless all

the constituent scenarios have proxies active with a set of spatial relations between them being satisfied. The proxies for the assembly-level constituents of an object-level scenario send location-labeled links to the summator of the object-level scenario. The summator unit checks that the appropriate spatial constraint (arms roughly above legs) is satisfied by active proxies; to the extent that it is not, the activation of the summator is limited.

We note in passing that the proxies could form the basis of a spatially-indexed hierarchy of action descriptions. Leg-actions and arm-actions are combined to form biped-actions. Similarly, two biped-actions (in different spatial locations) could be combined to form a multi-person scenario (e.g., a dance, or a common interaction such as meeting and shaking hands).

### 4.3.6 Summary

This chapter described the implementation of the architecture introduced in Chapter 2, using some of the mechanisms developed in Chapter 3. The lowest level transformation, form dot motion to segment motion, was not implemented because that problem has been previously addressed (see section 2.5.1) and our focus was on higher level processing. The next chapter describes a set of pilot experiments, and the subsequent chapter the main experiments.

# 5 Pilot Experiments

The architecture and implementation described in the preceding chapters is complex. In order to perform an initial test of the various representations and processing mechanisms, we gathered a small collection of data of real human gait, implemented the architecture, and ran simulations. The results were generally positive, and the conclusions led to a further set of experiments using more extensive data, reported in the next Chapter. In this Chapter, we describe the acquisition and analysis of the data used in the pilot experiments, the specification of features and construction of scenarios for the gaits, and provide the results of many simulation runs. We conclude with a discussion of the results and indications for further experiments.

## 5.1 Approach

For this set of experiments, gait data were acquired and analyzed as described in the next section. These data were used to design features to be used in the feature maps. They were also used to construct canonical scenarios representing the three gaits: walking, running and skipping. Scenarios for two transformations of walking were also constructed: inversion in vertical space (upside-down) and time (backwards). These transformations were intended to provide checks that superficially similar stimuli were not confused by the networks.

## 5.2 Data Acquisition and Analysis

The gait data were generated by having an actor[1] perform several examples of each gait in front of a fixed video camera. The actor moved parallel to the image-plane, consistent with our focus on 2D processing. Limb joints were marked with targets consisting of a white circle, with a black surround to maximize contrast (except for the hip, which was marked with a cross of reflective tape). The actor was illuminated and the film shot in a darkened auditorium. Figure 5.1 shows a single thresholded frame from a "skipping" sequence. As

[1] The author.

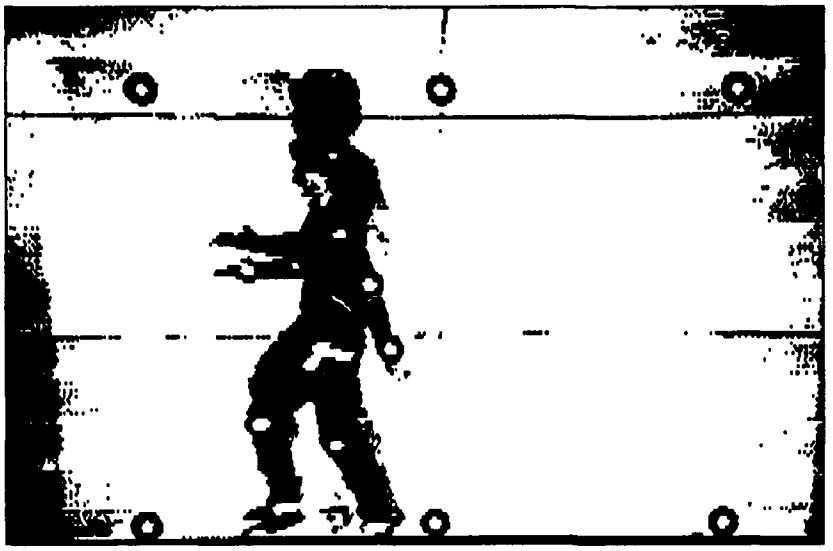

Figure 5.1: A Single Frame of Skipping

can be seen, the backdrop to the auditorium was quite reflective. Achieving a perfectly dark stage was never the intent, since we did not plan to perform the low-level image processing by computer. Instead, the criterion was that the targets attached to the joints be clearly visible when the tape of the sequences was replayed. The six fixed targets were to be used to register frames, but that was found to be unnecessary.

Once the images had been acquired on videotape, they were digitized it and software was written to display it frame by frame (or rather, field by field since there was motion in the scene) under the X Window system. Figure 5.1 is a bitmap dumped from the display. The Figure exaggerates the contrast; the actual display screen used 8-bit gray levels, while the Figure shows a one-bit deep bitmap. For each frame, the joint locations were noted by mouse-clicking over the targets, and selecting the name of the joint from a menu. The resulting raw data files contained the location of each visible joint for each frame in the sequence.

Software was written to animate these sequuences, so that they could be judged as adequate or not. They turned out to have a significant amount of jitter, which was smoothed out by noting the worst frames and editing the raw data files. Since the camera did not track the lateral translation of the actor, and since the field of view was not sufficient to capture an entire cycle of each gait, segments of different examples of each gait had to be spliced together to obtain complete cycles. During this process the translatory motion in the horizontal plane was zeroed out. Finally, the data were adjusted and smoothed by hand so that the final frame had the same position and velocity values as the initial one. This final frame was discarded, resulting in data sets of one complete cycle for each gait, which could be displayed over and over again to appear as, for example, walking-on-the-spot. This alignment of the beginning and end of the sequences was done so that more than one

cycle could be used as input to the network, and so that the input could begin at any phase of the cycle. After all these processes were complete, we were left with one sample of each of the three gaits: walking, running, skipping. The animated result was judged to be a good facsimile of the original video data.

Due to occlusion, the distal joints were often out of view. We required wire-frame data, i.e., data that showed all joints in each frame. This was achieved by discarding what distal data there were, and in their place substituting a copy of the proximal-joint data, but phase-shifted by 180°.

Finally, the wire-frame data were processed to determine the angle of the torso (with respect to the vertical), and the angle at each joint, for each frame. Together with the length of the limbs and the distance between shoulder and hip, this constituted a complete specification of each gait. The frame rate was 30 frames/sec. A cycle of skipping took 36 frames or 1.2 seconds; walking, 30 frames or 1 second; and running, 24 frames or 0.8 seconds.

## 5.3 Feature and Scenario Design

Apart from the representation of segment-level features, specification of features and scenarios is performed in software, using a symbolic model of a biped. Although below we anthropomorphize the description of the construction of features and scenarios, using "we" instead of "the algorithm", the reader should remember that none of this tedious work is done manually. Rather, given the gait data, the network construction program defines both features and scenarios automatically. The gait data files do not contain motion information, but the network construction software computes the angular velocity of each segment in each frame as it reads in the gait data.

### 5.3.1 Segment Level Features

Recall that the segment level represents line segments. The segment level features were designed with two criteria in mind: they should be knowledge-independent (i.e., not associated with knowledge of particular objects in the world); and their resolution of spatial information and motion information should be coarse. The reason for the former is that segments are at a sufficiently low-level that it is reasonable to expect a representation that spans the space of all theoretical possibilities. For example, line segments can occur at any orientations, so the representation should be able to represent a line segment at any orientation. The reasons for coarse resolution are twofold. First, the finer the resolution the greater the number of units and links needed for representation and for using the representation. Second, it is unreasonable to expect that fine resolution information is necessarily available from the lower levels (e.g., the structure-from-motion process). Accordingly, shape features at the segment level are coded by orientation (in 30° increments) and length (three lengths:

short, medium, long). Thus there are 36 different segment-level shape features (12 orientations $\times$ 3 lengths). Motion features are coded by orientation of the line segment (in 90° increments) and angular velocity (5 quantizations: fast anticlockwise, slow anticlockwise, stationary[2], slow clockwise, and fast clockwise). Thus there are 20 different segment-level motion features (4 orientations $\times$ 5 velocities). Location is represented by replicating the representation at each different location in the hexagon-based tiling described in Chapter 4.

### 5.3.2 Component Level Features

Recall that component level features represent leg-like and arm-like combinations of line segments. At this level, the features are assumed to be knowledge-based, i.e., we represent only those combinations of line segments that actually occur in the world, rather than every theoretically possible combination. This is an important distinction. At the segment level, there are 36 different shape features and 20 different motion features. Component-level shape features (in our case) consist of two segment-level shape features, arranged in a particular spatial configuration. We discuss spatial configurations below; for now, let us simply state that there is the potential for 48 different configurations for shape features and 24 different configurations for motion features. This means we have the possibility of $48 \times 36^2$ different shape features and $24 \times 20^2$ different motion features, or approximately 72,000 different features at the component level. In fact, few of these combinations occur in our input data and we represent only those combinations that do occur, and some of these combinations are collapsed into a single component-level feature as described below.

#### Spatial Composition of Component-Level Features

Each component level feature consists of one segment that occurs at the same location as the component (e.g., the thigh occurs at the hip, which is also where the leg occurs), and one segment that occurs some distance away (e.g., the calf occurs at the knee). A component-level feature consists of a *local* segment and a *remote* segment in a particular *remote-region*. For component-level features, the remote-regions are defined by the area which is both between two concentric circles and also inside a cone whose apex is at the center of the circles. Figure 5.2 illustrates such a region for a leg-like feature. The feature represents every combination of segment-level features in which there is one (or more) local segment-level features of the correct type (within the component location), and one or more remote segment-level features of the correct type (within the shaded region). Refer to Figure 4.5 for the network structure used to implement this.

For shape features, the angle of the cone is 20° arranged in an overlapping fashion with the cone axes every 15°. For motion features the cone angle is 40° and there is a different cone every 30°. The overlap is to ensure complete coverage of visual space given the discrete hexagonal tiling used. For each type of feature, there are two sets of radii pairs

[2] In actuality, this represents a range of small angular velocities either side of zero.

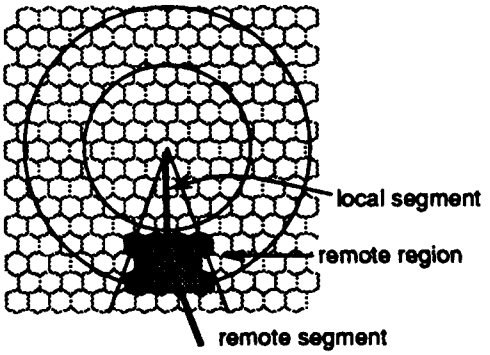


Figure 5.2: A Region for a Component-Level Feature

(i.e., two sets of concentric circles) which define regions near and far from the center. These numbers mean that there are $2 \times 24$ different shape regions and $2 \times 12$ different motion regions.

### Collapsing Motion Features at the Component Level

In addition to only representing those component-level features that actually occur in the world, we merge some of the motion features together. Whenever a motion feature is constructed, it automatically represents not only the specific combination of local segment, remote segment and remote region that gave rise to it, but also the three other combinations that differ only in the orientation of the remote segment. This reduces the number of component-level motion features by a small amount and, more importantly, it emphasizes differences in the motion of the local segment over differences in the remote segment.

The result is that we need only 85 shape and 46 motion features at the component level. Were a wider range of gait data available, some additional features would no doubt be needed. But the main point is: that of all the possible things that *could* occur in the world, very few actually *do*, and we need represent only those few that do.

## 5.3.3 Assembly Level Features

Features at the assembly level are composed of two component-level features, thereby representing "pair-of-arms" or "pair-of-legs" in different configurations that can occur. Once again, we only represent those combinations of component-level features that actually occur in the gait data, rather than all the combinations that are theoretically possible, requiring 101 shape and 75 motion features. The two constituent features must occur at the same or adjacent component-level locations (as illustrated in Figure 4.6).

### 5.3.4 Assembly Level Scenarios

In order to construct a scenario, we must specify the events and the time intervals between their onsets. An event consists of one or more enabling conditions which indicate its onset, and zero or more supporting features which sustain it until the next event. Once the phase of each event in the gait cycle is decided, the time intervals between them are determined. Separate scenarios are constructed for the different assemblies: in our case "pair-of-arms" and "pair-of-legs"; and for the different gaits. Thus, at the assembly level we have twelve scenarios (two assemblies $\times$ three gaits $\times$ two directions).

The major principle behind scenario design is that event onsets should be determined by significant changes in motion. The way we construct a scenario is as follows. We examine the gait data, frame by frame, until we find a significant change in motion. A new event is specified, with its enabling condition being the significant change that was found, and its supporting features being all those that occur until the next significant change. The interval between one event and the next is the number of frames that occur between the detection of the respective enabling conditions. This process repeats until the last item of gait data is reached. The features that occurred before the first event are added as supporting features of the last event. Finally, the intervals between the last event and the first is determined, completing the cycle.

A significant changes in motion is found whenever a segment-level feature representing the local segment of one of the components of the assembly changes. For example, this means that whenever the thigh undergoes a change in motion sufficient to put it over a segment-level motion boundary, a new event commences in a "pair-of-legs" scenario. Likewise, events in the "pair-of-arms" scenarios are determined by the motion trajectory of the upper arm. The reason for selecting this rule is discussed in the section on structure-from-motion in the final Chapter; to oversimplify, it is that we expect the local segments to be recovered more reliably and more quickly by the structure-from-motion process, because if the imaging system is tracking the cloud of dots then the local segment pivot points (hip and shoulder) are approximately stationary[3].

### 5.3.5 Object Level Scenarios

Object level scenarios are the output of the system. They represent entire gaits, such as "biped-walking". Their events are composed of events from one or both of two assembly-level scenarios, e.g., "arms-walking" and "legs-walking". The idea is that there be fewer events at the object level than at the assembly level, and that an object-level event should group temporally adjacent assembly-level events. The rule for deciding which assembly-level events are combined into one object-level event is that for any object-level event, its constituent assembly-level events have onset times that are within two frames of each other; and that the onset times of consecutive assembly-level events are the same or one

[3]Circles are easier to find than epicycles.

frame apart. Given the two constituent assembly-level scenarios, there may be more than one object-level scenario that satisfies the rule. We arbitrarily choose the first one we find.

## 5.4 Network Simulation Details

In the simulations, the gestalt mechanism designated to be used in the feature hierarchy (O-binding, see Chapter 3) was not used. However, its extension (P-binding) designed to deal with forming gestalts between the spatially indexed feature hierarchies and the central scenario representations was used. The reasons for this choice were that the O-binding mechanism is not new, and has been shown to work [Olson, 1989], and the architecture is complicated enough that stripping unnecessary mechanisms would ease the job of elucidating the contribution of the remaining processes. P-binding, on the other hand, was a new and untested idea, and its basis, the proxy units, were considered to be one of the most interesting developments to have arisen in the architectural design phase. At this stage of the work, the attention mechanism (which focuses a scenario on certain locations) had not been developed and so was not used. In addition, the proxies were not arranged to compete, another essential aspect of the attention mechanism. Feature priming by proxies and interval units was used.

The simulation step was set at 1/300 second real-time equivalent, derived from a frame rate of 30 frames per second and 10 network updates per input frame. There is nothing special about these particular values other than video frame rate is 30 frames per second and 300 update steps per second is a reasonable approximation to neural switching rates. Synchronous update was used[4]. Presentation of a frame was performed by turning on the segment-level shape and motion features representing that frame. These input units remained active for the next 10 simulation steps, after which they were zeroed and the next frame presented.

## 5.5 Experiments

Some experiments varied the spatial scale of the input, others varied the tempo, for example by allowing only 9 simulation steps between frames (a 10% speedup). In some experiments two gaits were presented at once in overlapping or separated locations, in others the location of the biped was translated across the visual field at approximately the actual speed of the gait. Static background clutter consisting of all the segment-level shape features at random locations was applied in some cases (one can think of overlapping gaits as representing an extreme case of dynamic background clutter). Presentation of frames in random order was also tried. The input was presented at different visual field locations and with different

[4]In synchronous update, each unit examines its inputs and computes its output. When all outputs have been computed, and not before, the new output values are made visible to the other units.

initial phases in different trials, but exhaustive tests of each different gait sample presented in every possible permutation of spatial scale, tempo, combination, clutter, location and phase was clearly out of the question. In the sections that follow, we describe the experiments and show representative results.

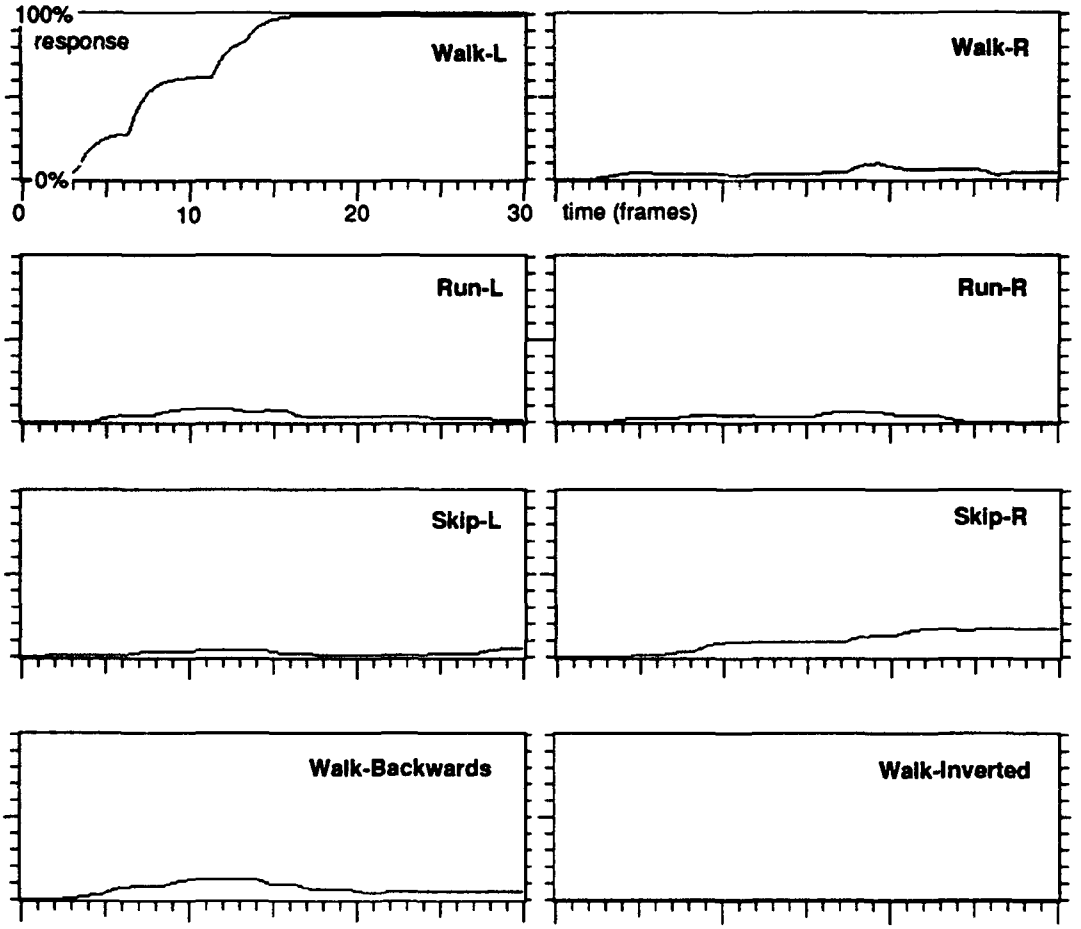


Figure 5.3: Canonical Walk-Left Simulation

### 5.5.1 Experiment 1: Canonical input

In Experiment 1, the each of the three gaits was presented to the network, one at a time, with varied starting phase and location in the visual field. The input data, therefore, were precisely the same as that used to construct the features and scenarios. This experiment was intended to verify that the network could discriminate perfect input, for various phases and locations. A plot of the result of a representative simulation run is shown in Figure 5.3. The left hand column shows the output of the summation unit for the object level scenarios representing walking, running and skipping to the left, and the temporally-reversed transformation of walking left. The right hand column shows the outputs for the same gaits in a rightward direction, plus the upside-down version of walking left. The network requires just over 15 frames of input in order for the correct scenario to reach full activation, and none of the others ever reach a significant level of activation. The step jumps in activation for the target scenario correspond to the firing of the event units in the scenario. In this case,

the target reached full activation after approximately four events. This experiment showed that the network could discriminate each gait from the other two, and from the following transformations of the gaits: mirror-image, upside-down, and reversed in time.

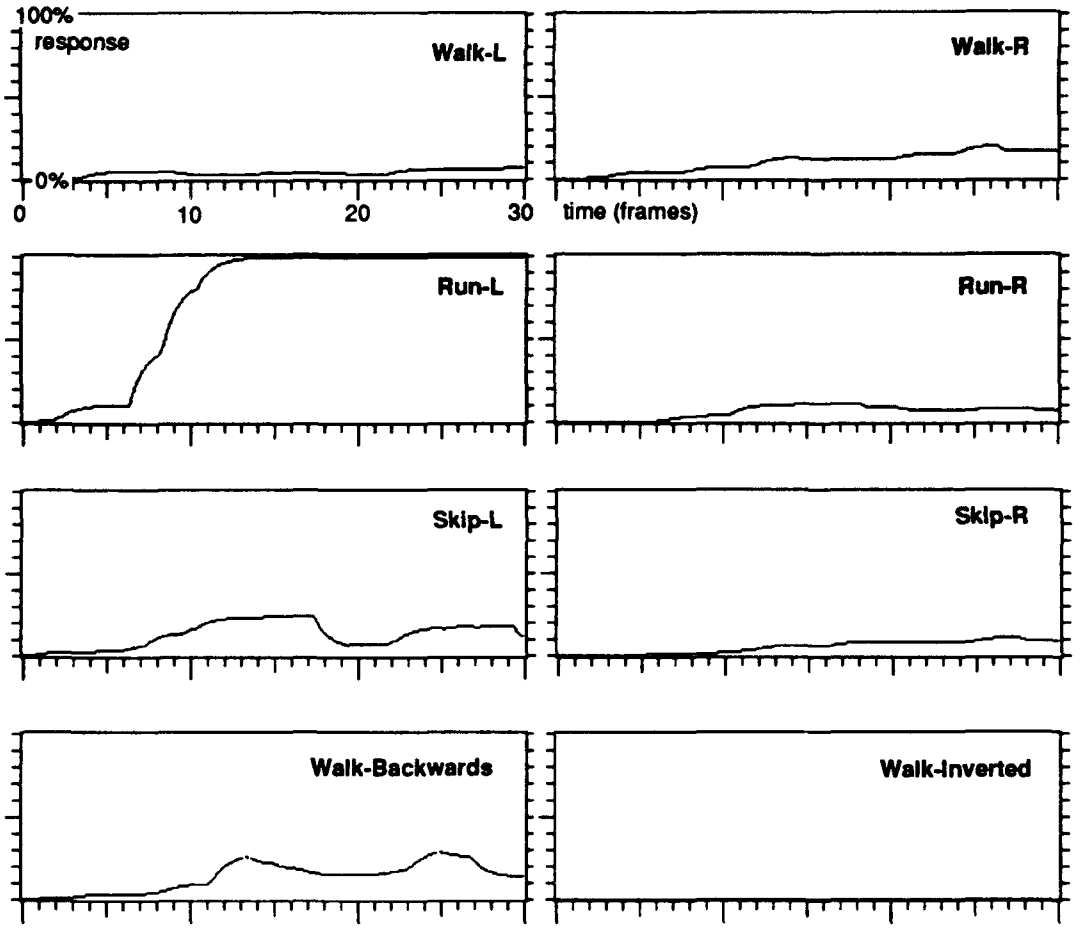


Figure 5.4: Simulation of Running-Left at 3/4 Scale

### 5.5.2 Experiment 2: Size variation

In Experiment 2, we varied the size of the biped. We investigated magnification factors of between 0.5 and 1.5. Figure 5.4 shows the output from a representative simulation run, in which running left was presented at a magnification factor of 0.75 (i.e., three-quarters of the normal size). We found that there was tolerance of magnification factors from about 0.65 up to 1.5. This tolerance was designed for by having very coarse quantization of length in the segment-level features and in the coding of remote regions. The experiment showed invariance to significant spatial scale transformations; however, the numbers reflect the quantizations chosen in the implementation rather than anything more fundamental in the architecture. Magnification factors smaller than about 0.65 resulted in insufficient separation of arms and legs in the attention map, and factors larger than about 1.5 resulted in stimuli extending well outside the visual field.

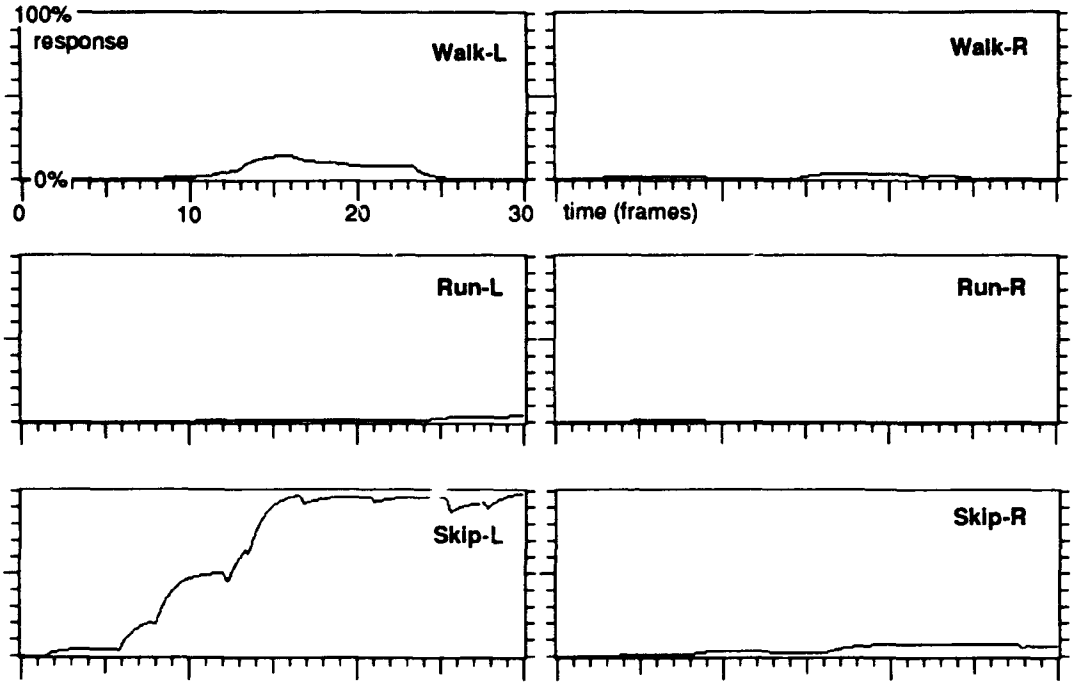


Figure 5.5: Simulation of Skipping-Left with 10% Slowdown

### 5.5.3 Experiment 3: Tempo variation

In Experiment 3, we investigated tolerance to different tempos. The tempo was increased or decreased by changing the number of simulation steps between frames. Since the representation of time in the network is in terms of the number of simulation steps, this is a rough approximation to changing the tempo. One important aspect of motion that it fails to capture is that for different tempos, the velocities are different. However, for small changes in tempo the velocity change is small. We expected the networks to be fairly sensitive to tempo change because of the fixed nature of the time intervals represented in the scenarios. Velocity effects were expected to be of less significance, at least for small variations in tempo, because the coding of angular velocity at the segment level is very coarse.

Figures 5.5 and 5.6 show the scenario activations when the tempo decreased by 10% and 20% respectively. This was achieved by setting the number of simulation steps per frame to be 11 and 12 respectively, instead of the usual 10 steps per frame. From this point on we omit the trace of the activation of the temporally and spatially inverted scenarios because they never became significantly active in any of the experiments.

This experiment showed that there is tolerance to small variations from the canonical tempo, but that performance rapidly degrades as the variation from the canonical tempo increases. We attribute what tolerance there is to the ramp up and down in the priming provided by the interval units. Recall that as a scenario becomes more active, these ramps

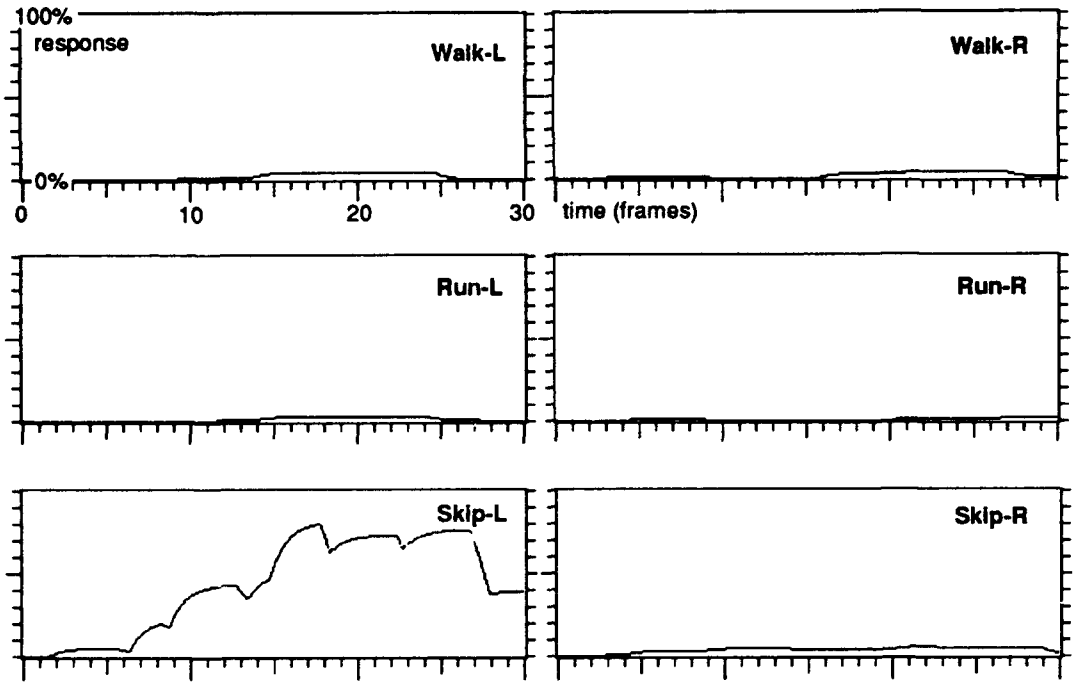


Figure 5.6: Simulation of Skipping-Left with 20% Slowdown

become steeper. In Figure 5.6, the skipping-left scenario becomes highly active and then loses activation. This is because the temporal constraints are initially very lax (ramps have a shallow incline) and are therefore satisfied by the input; the scenario activation builds up. But once the scenario is highly active the temporal constraints are strict (ramps are steep) and are no longer satisfied by the input; scenario activation drops. In the final Chapter, we discuss our ideas for determining the tempo dynamically. If it can be done, then the scenario representation of time (in the interval units) can be scaled using the derived tempo.

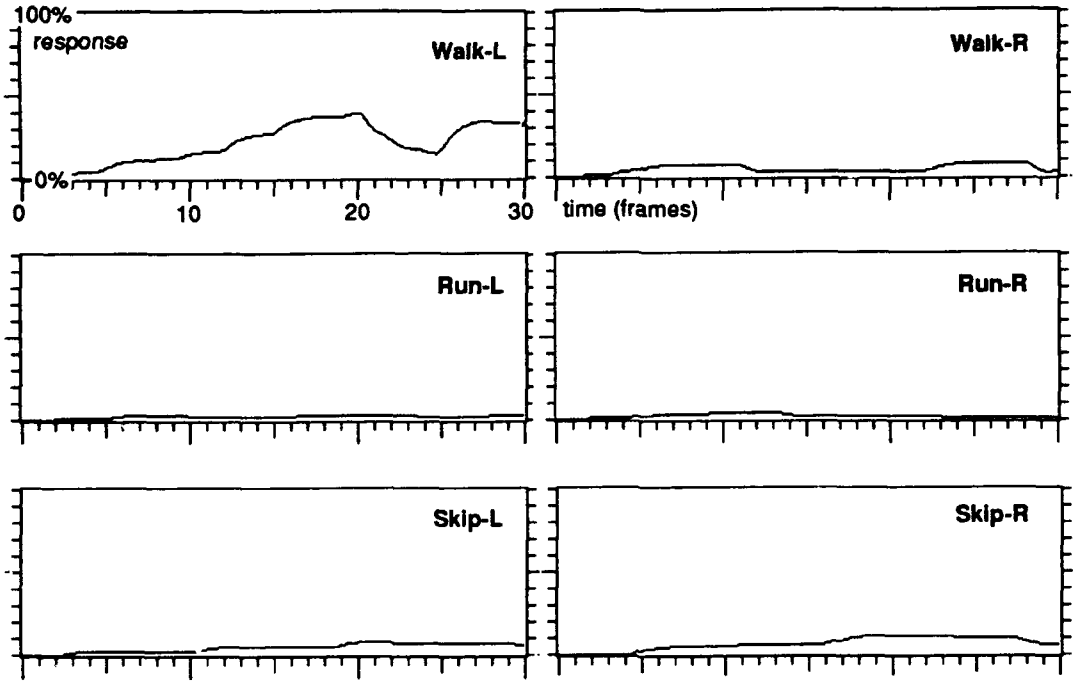


Figure 5.7: Simulation of Walking-Left with Random Frame Order

### 5.5.4 Experiment 4: Random frame order

In Experiment 4, we tested the response of the network to frames presented in a random order. This is not quite the same as splicing random frames from the videotape together, because each of our input frames contains motion information as well as spatial information. The intention was to check that the network could discriminate incorrect sequences from the correct sequence, even if the elements in the incorrect sequence were all drawn from the correct sequence. Figure 5.7 shows the results of a simulation run in which the frames from walking-left were presented in random order. As can be seen, the walking-left scenario becomes significantly more active than the other scenarios, but it barely reaches 40% activation at any time. The results indicate that correct order of presentation is indeed important for achieving significant levels of activation in a scenario.

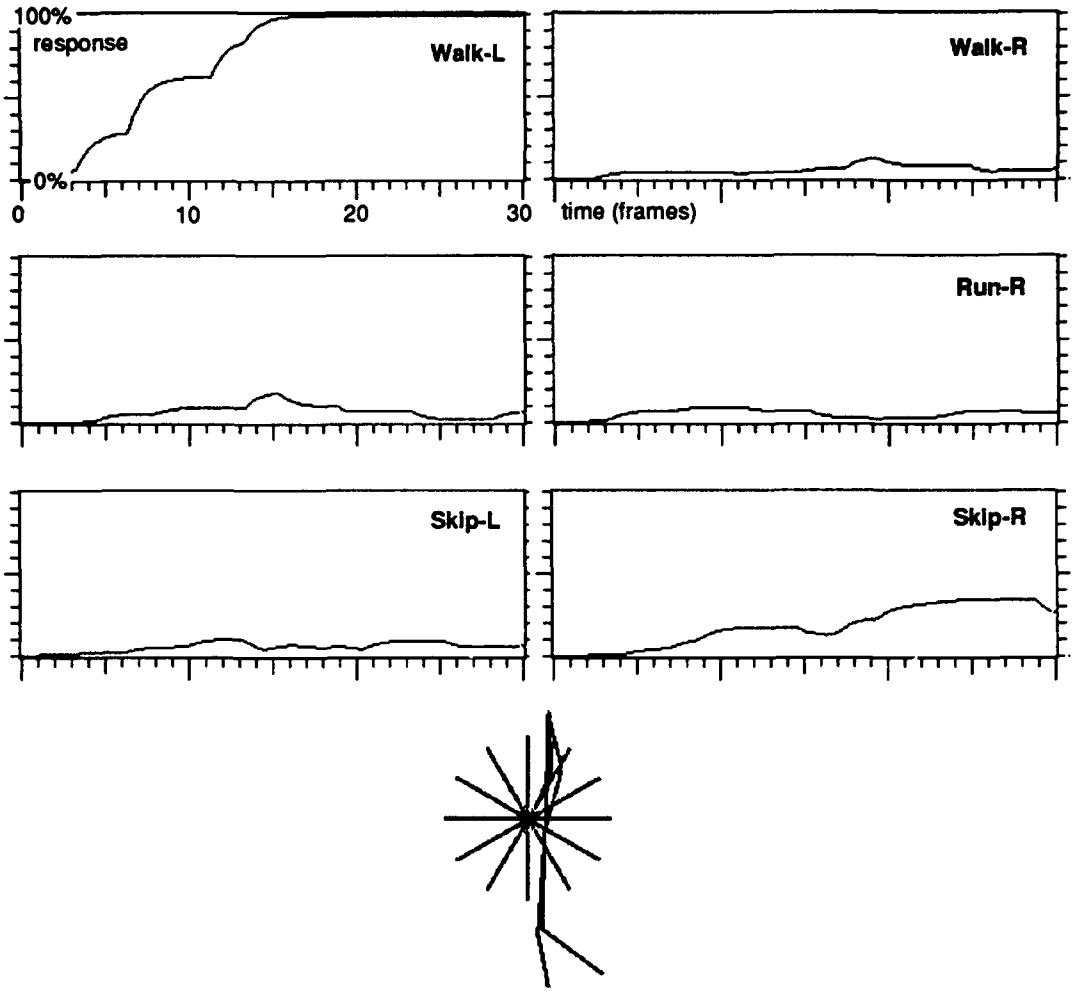


Figure 5.8: Walking-Left with Static Clutter

### 5.5.5 Experiment 5: Static clutter

In Experiment 5, we investigated the effect of static background clutter. Clutter was simulated by turning on all the segment-level shape features at one or more locations in the visual field. Figure 5.8 shows a representative result when clutter was applied at one location close to the hip of the biped (an important location for the network). Figure 5.9 shows the result when clutter was applied at 8 locations overlapping and near to the biped. There is little effect on the activation of the target scenario in either case, although some interference occurs with in the heavily cluttered scene. However, in the heavily cluttered case, the skipping-right scenario achieves significant levels of activation. Even so, the target scenario is clearly the winner. The fundamental reason that so much static clutter is tolerable is that the scenarios look at motion information for event onset, and because the scenarios integrate over time. Thus, static information is of little importance - in fact, it

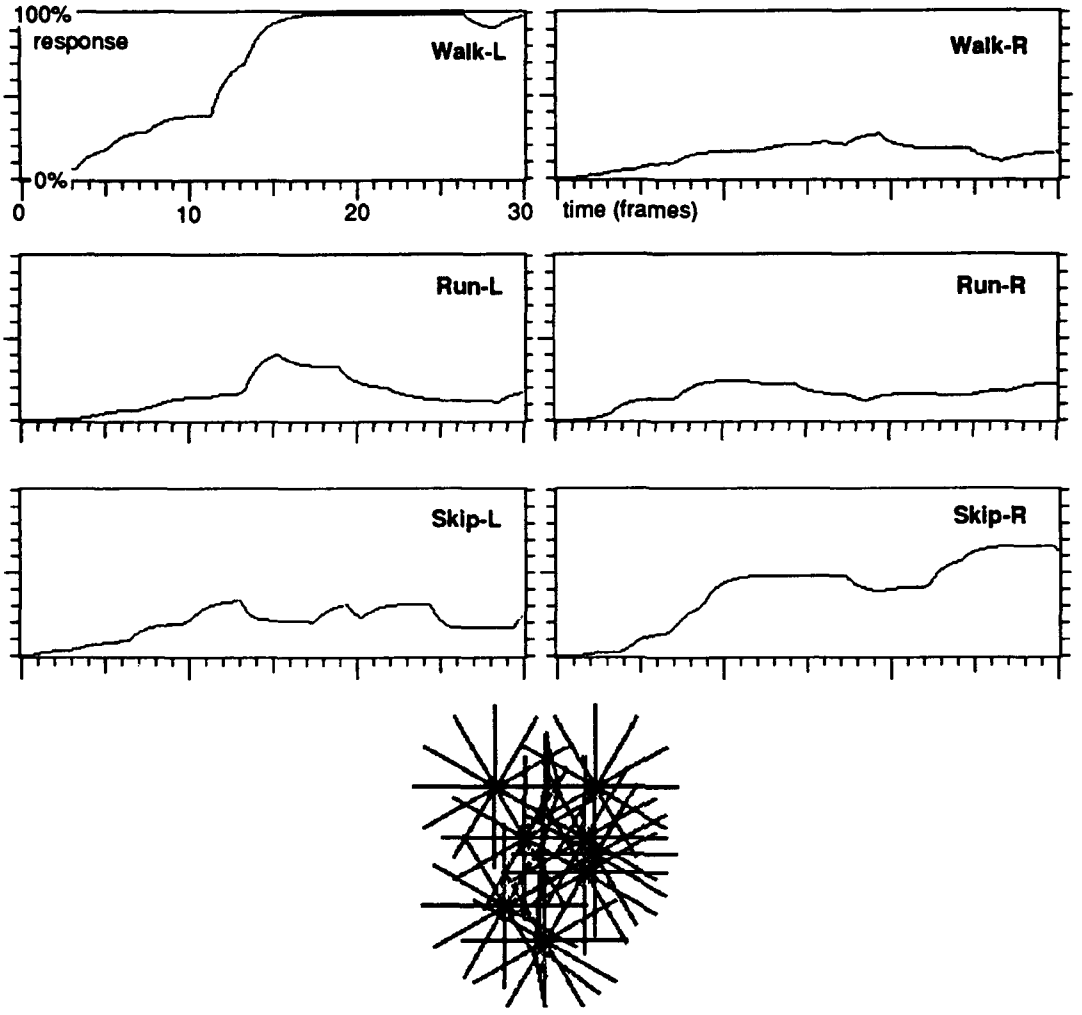


Figure 5.9: Much Static Clutter

is only used as supporting evidence for an event already enabled by a change in motion. The positive result even in the heavily cluttered scene is not representative of the ability of a more complete vision system. We would expect that the lower level processes which recover the line segments would be highly confused and would not even be able to recover the moving line segments necessary for the result we demonstrated. What this experiment does show is that so long as the correct motion information can be recovered by the lower level processes, it does not matter if significant numbers of irrelevant shape features are also activated, be they veridical or not.

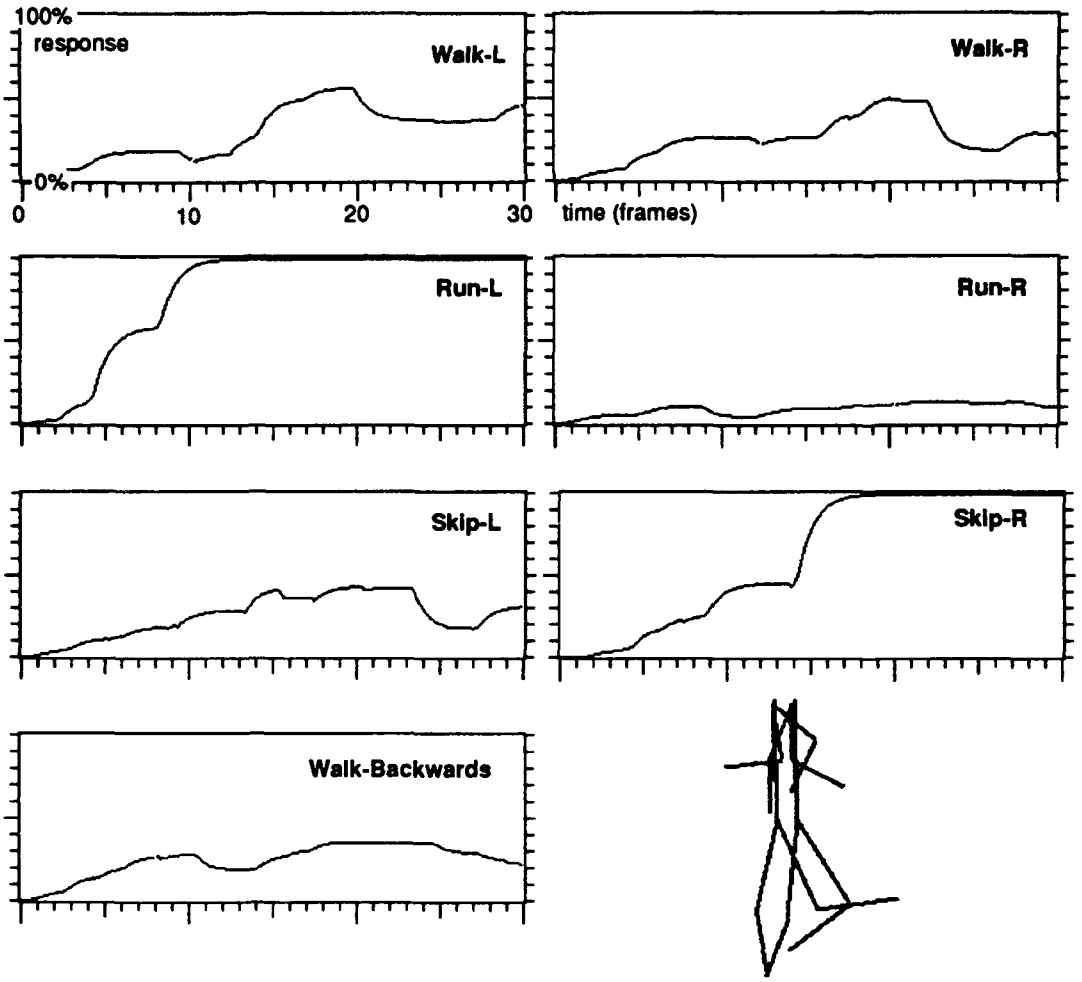


Figure 5.10: Run-Left and Skip-Right Overlapping

### 5.5.6 Experiment 6: Dynamic clutter

In Experiment 6 we investigated the effect of dynamic background clutter (that is, background motion). A simple way to generate background clutter is to present two gaits at once. If either gait is designated target, then the other is clutter. This is an extreme form of clutter, since the spatial structure of the clutter, and to some extent its motion structure, is very similar to that of the target. The clutter results agree with perceptual data[5] [Cutting *et al.*, 1988] in which it was shown that a static mask had almost no effect on discrimination, whereas dynamic masks (particularly those composed of pieces of the target gait) reduced ability to discriminate quite markedly.

[5]Clutter is a rough parallel to the concept of a "mask" in perceptual psychology experiments.

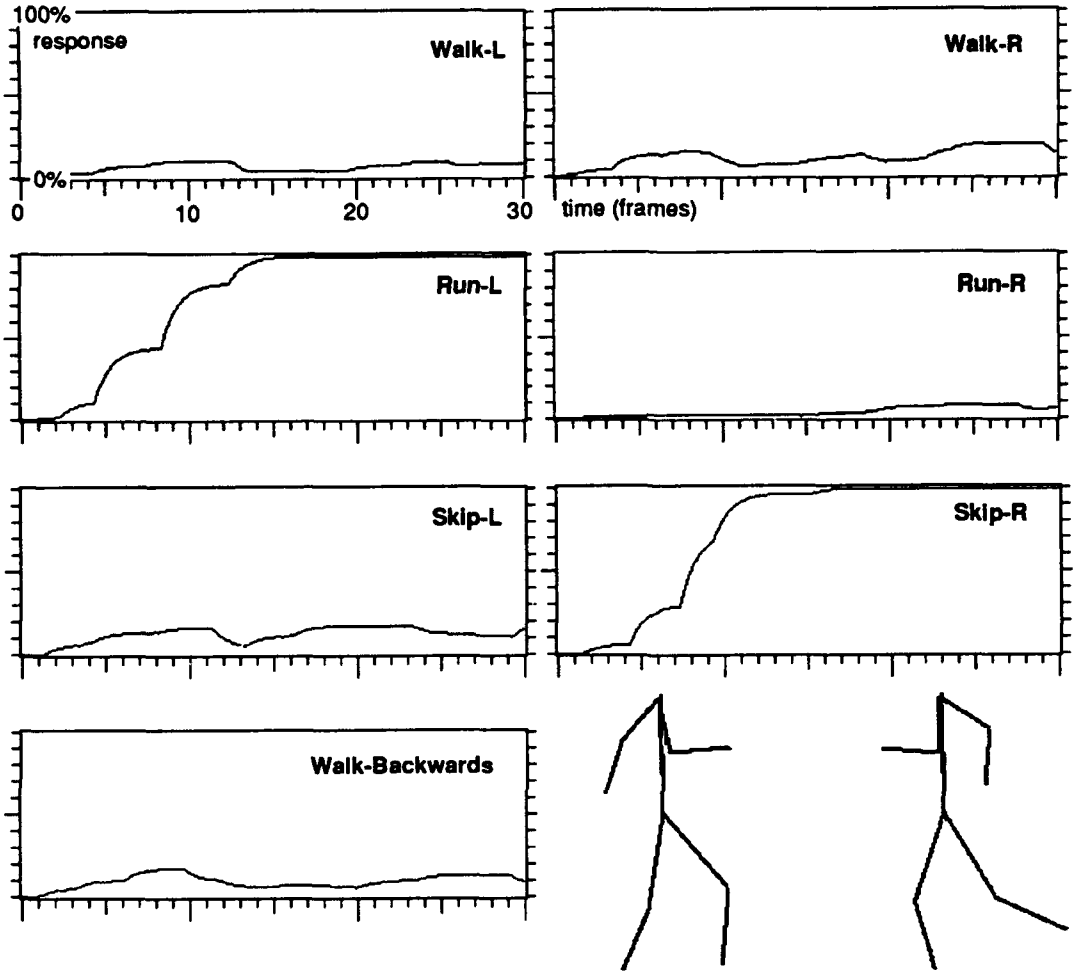


Figure 5.11: Run-Left and Skip-Right Separated

### Spatial Proximity

If the two gaits are presented in close proximity, then the likelihood of illusory shape and motion features being activated is high. In this case, we expected poor performance. On the other hand, if the two gaits are widely separated in space, then no such illusory conjunctions occur. In addition, the provision in the design of location-specific sites on the event units was intended to minimize crosstalk between features at different assembly-level locations. Moreover, the proxy mechanism allows different scenarios to compete for the activation from particular locations. These three considerations led us to believe that performance would be good if the clutter was separated in space from the target.

### Overlapping Clutter

Figure 5.10 illustrates the result when two gaits are presented simultaneously and in neighboring visual locations. The target scenarios, run-left and skip-right, are activated correctly within about 15 frames. But there is also significant activation in some of the other scenarios. We do not show the simulation traces, but performance is worse if three different scenarios are presented simultaneously in adjacent segment-level locations. As expected, overlapping clutter results in poor performance.

### Non-Overlapping Clutter

Figure 5.11 illustrates the result when two gaits are presented simultaneously and but in distinct assembly-level locations. The target scenarios, run-left and skip-right, are activated correctly within about 15 frames, and none of the other scenarios achieve significant activation. As expected, dynamic clutter, even of a similar structure to the target, results in little degradation if the clutter and the target are widely separated in space.

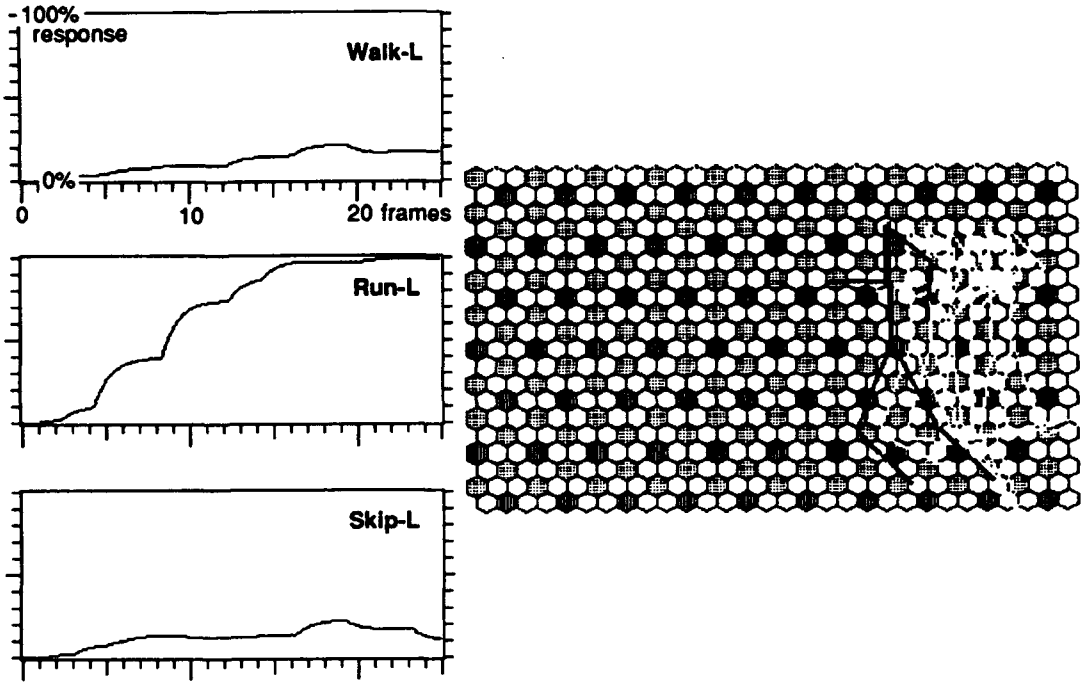


Figure 5.12: Run-Left with Translation

### 5.5.7 Experiment 7: Translation

In Experiment 7 we tested the network performance when translatory motion was added to the gait. This can be thought of as viewing a figure moving through the scene rather than a figure moving in place. Although the design had assumed that the moving figure would be tracked by the imaging system, therefore remaining essentially in the same visual location, the only part of the architecture that actually depends on this assumption is the priming of features by the combination of proxy and interval units. An active interval unit will prime those locations where there is an active proxy; but if there is translation, the figure may have moved on to a different location in the meantime. Since this priming is of most use in speeding up response and in providing feedback to lower levels, and since the proxy (assembly-level) locations have significant overlap, we did not expect the addition of translatory motion to affect performance much.

Figure 5.12 shows the result when translatory motion was added to the run-left gait. The visual field is shown in the figure: hexagons show the segment-level locations; shaded and striped hexagons show the component-level locations; and striped hexagons show the assembly-level locations. To simulate translation, we shifted the figure left one segment-level location per frame. This was approximately the same rate as appeared in the original video data. One aspect it fails to simulate is that all the absolute velocities of the line segments have the translation velocity added to them. However, the angular velocities of the segments, which is what our motion features are based on, do not change with addition of

a translatory component. The size of the network required for this simulation was such that only three of the scenarios were included. The simulation trace shows a slight slowdown in response by the run-left scenario (compare with Figure 5.4), and negligible activation in the other two scenarios. This experiment shows that the architecture can discriminate an untracked gait almost as well as it can discriminate the tracked gait.

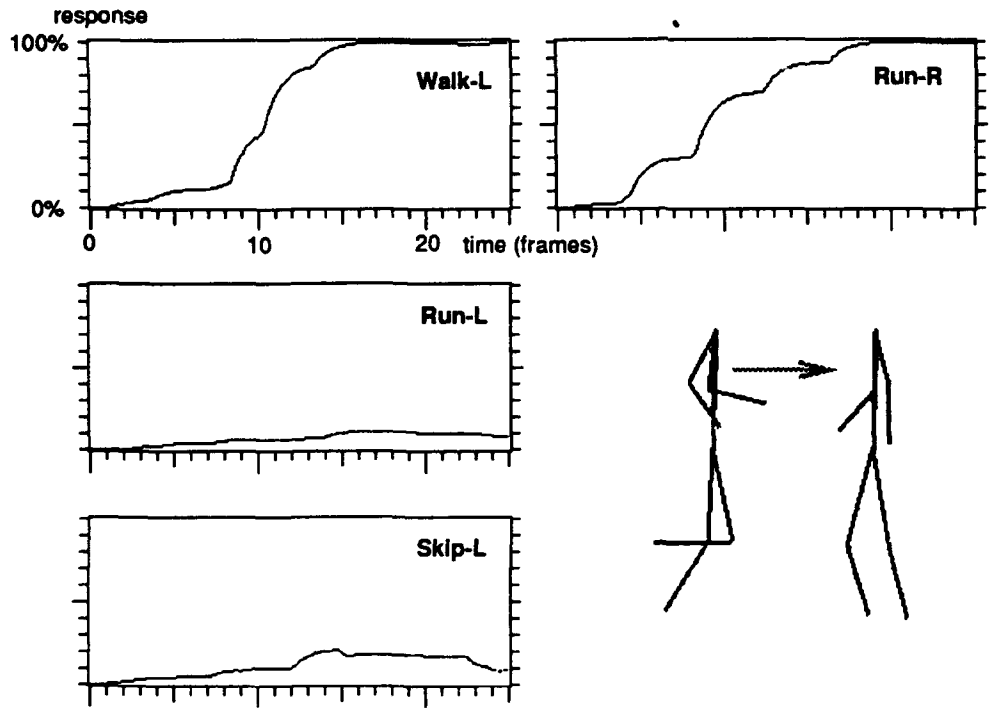


Figure 5.13: Walk-Left Fixed and Run-Right Translating

### 5.5.8 Experiment 8: Translation with Transitory Clutter

Given the results of Experiments 6 and 7, we decided to test the network on a combination of tracked and untracked gait. We presented one gait in a fixed location, and another gait translating from one side of that location to the other. This can be thought of as tracking a person walking, while another person runs past the tracked person in the opposite direction. Experiments 6 and 8 led us to believe that the two gaits should be discriminable while spatially separated, but that there might be confusion during the period when the translating gait was in close proximity to the fixed gait. Given the integration of activation over time in the scenario networks, we expected the confusion to be small. Figure 5.13 shows the simulation trace when walk-left (tracked) and run-right (translating) were presented simultaneously, with run-right initially on the left of the visual field and walk-left in the middle of the field. The trace shows that there is very little negative effect on the activation of the walk-left scenario, and the run-right scenario achieves full activation within 20 frames. This experiment shows that the scenarios are tolerant of transitory background clutter even if the same clutter causes serious problems when it is continually present.

## 5.6 Discussion

The results of the pilot set of experiments showed the power of the design to discriminate gait even with significant transformation or cluttering of the input data. The following variations on the canonical input were investigated:

- **Spatial Scaling**: tolerance to spatial scaling was shown, and it was attributed to the coarse quantization of spatial attributes of the features.
- **Tempo**: sensitivity to tempo was found. Small variations ($\pm$ 10%) from the canonical tempo were tolerated, and this was attributed to the ramps in the interval unit transfer function.
- **Random Order**: sensitivity to the order of the sequence was found. Presentation of the sequence elements in random order did not produce high levels of activation. This was attributed to the explicit sequence in the scenario representation.
- **Static Clutter**: tolerance to large amounts of static clutter was shown. This was attributed to the emphasis on motion features and correct sequence for activating scenarios.
- **Dynamic Clutter**: tolerance was shown if the clutter was spatially separated from the target and this was attributed to the location-specific sites on the event units and to the proxy mechanism which allows a scenario to "own" a location. Persistent clutter overlapping the target was shown to produce false positives but not false negatives, and this was attributed to the high incidence of illusory conjunctions. However, tolerance to transient clutter overlapping the target was shown, this being attributed to the hysteresis in the integration of activation in the scenarios.
- **Translation**: tolerance to realistic translation was shown, and this was attributed to the fact that the only aspect of the architecture that assumed a fixed-location input was the priming of features by the scenario, and given the high degree of overlap in the assembly-level tiling, even this priming process works most of the time.

The area in which one would particularly like to do better is that of dealing with variations from canonical tempo. It is a fact that many structured sequences which biological perceptual systems have to deal with can have tempos which vary from the canonical by substantially more than 10%, e.g., visual movement recognition, speech-recognition and music processing. Dealing with arbitrary tempo is not necessary, but dealing with variations from canonical tempo by a factor of 2 in either direction would seem to be a minimal requirement. The scenario representation can easily handle variations from canonical tempo if an accurate tempo measurement is available. The hard question is how to determine tempo, especially an initial good estimate. Although we developed some ideas towards a solution, we decided not to pursue them here. Our ideas for a line of research appear in the final Chapter.

### 5.6.1 Indications for Further Work

Two major weaknesses of the experiments described in this chapter are:

1. **Limited Data.** The data used to generate the features and scenarios were extremely limited - one example of each gait. Furthermore, the exact same data were used to test the networks, albeit transformed in some systematic way.
2. **Contribution of Specific Mechanisms.** The experiments did not go far enough in elucidating the role played by each processing mechanism, either in improving speed or disambiguation of response, or in enabling the architecture to discriminate otherwise indiscriminable inputs.

In addition to these weaknesses, there was no systematic study of:

- how initial phase of presentation affects the response.
- if transition from one gait to another is handled successfully.
- if discrimination improves or degenerates with presentation of multiple cycles of input.

and there was no control for discrimination based on cycle time (recall that cycle times are 0.8 seconds for running, 1.0 seconds for walking, and 1.2 seconds for skipping). To address all these weaknesses, we decided to collect a wider range of gait data and conduct a second set of experiments.

# 6 Main Experiments

The aim of the second set of experiments was to address the weaknesses of the pilot set. First, we gathered multiple samples of multiple individuals performing each of the three gaits, and verified the architecture using this more extensive set of data. Second, we clarified the contribution of each of the different processing mechanisms. Finally, we investigated some extra questions that had not been addressed in the earlier study: how initial phase affects response; transition from one gait to another; and the response when multiple figures are present in the input data.

## 6.1 Approach

We decided to remove the shape features from the architecture. The shape features were discarded because it was felt that they were not necessary for satisfactory discrimination (and the motion features do include some spatial structure). We also added switches to turn off construction and/or simulation of several mechanisms. After reviewing the architecture and the results of the first set of experiments, we realized that the proxy units were much more useful than originally thought, and we formulated the attentional mechanism described in Chapters 3 and 4. We normalized each of the gait samples to 60 frames/cycle, to remove any question that discrimination could be based on cycle time. The gait samples were all of movement from left to right. We reflected the data about a vertical axis to produce samples of movement from right to left.

## 6.2 Data Acquisition and Analysis

We were fortunate to be offered the use of an image-processing system specifically designed to gather data on human gait. The WATSMART system [Scholz, 1989] consists of two cameras arranged to give a stereo view, a set of light-emitting diodes (LEDs), a calibration frame, and an IBM RT with software for processing the camera signals. The LEDs are attached to the actor with surgical tape, and a "tail" of wires from the LEDs return to the

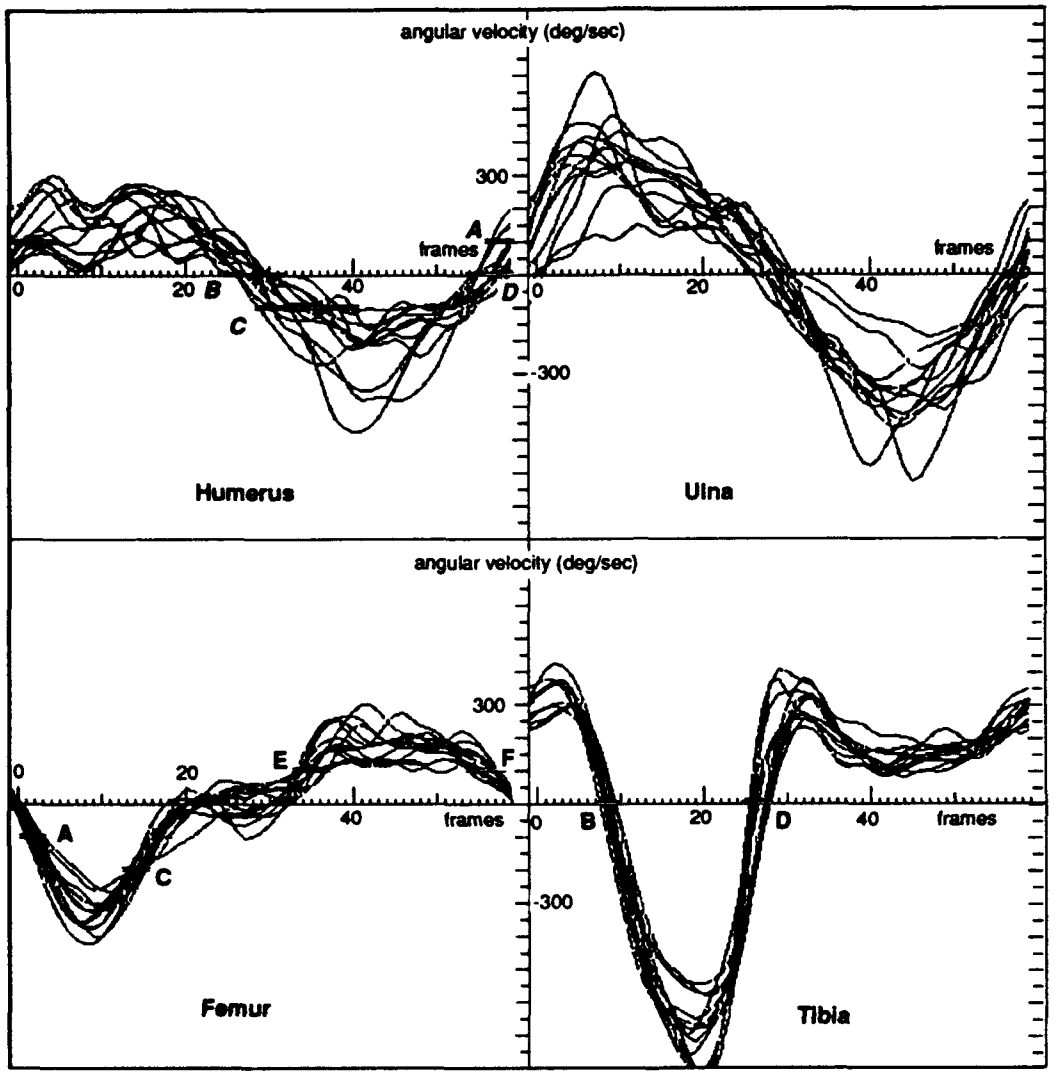


Figure 6.1: Angular Velocity of Limb Segments for Walking Data

computer. The system gathers data as the actor moves by sequentially flashing each LED and recording the location of the LED in the camera frames. The system can gather data from 8 LEDs at up to 400 frames/second. Software then computes 3D location of each LED in each frame by examining the 2D frame data from the two cameras. The system is calibrated with a rigid cuboid containing LEDs at the vertices. We operated the system at 100 frames/second, with LEDs attached to the six proximal joints (shoulder, elbow, wrist, hip, knee, ankle) and the two distal joints which were most often visible (wrist and ankle). Actor motion was roughly perpendicular to the calibration axis (average of two camera axes).

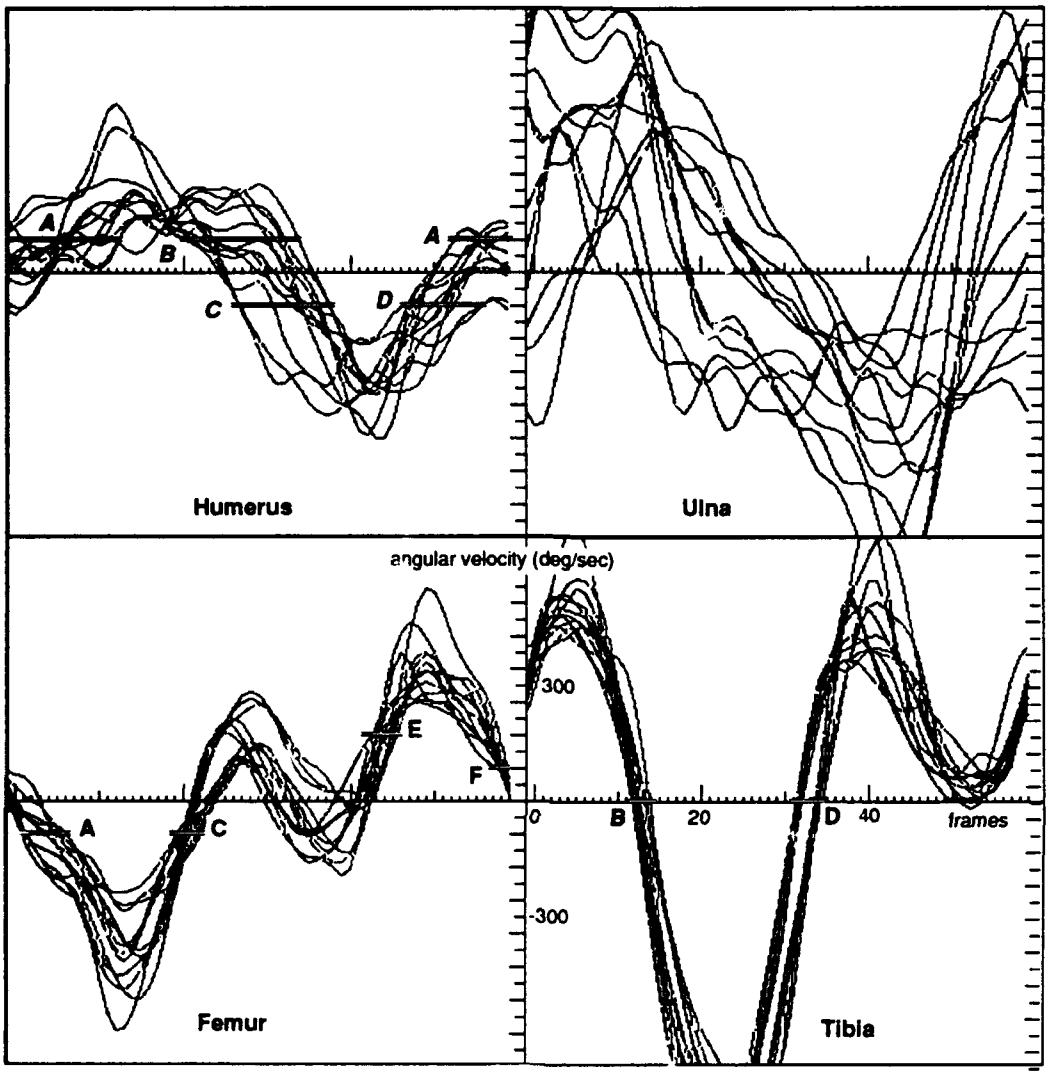


Figure 6.2: Angular Velocity of Limb Segments for Running Data

We gathered data from four actors[1], two male and two female. We collected three samples of each gait for each person. Interestingly, one of the males had never skipped in his life and had no idea what the gait was like, but was nevertheless familiar with the term! Therefore, we collected skipping data for only three of the four actors.

The raw data files containing 3D location (to within millimeters) of each (labeled) LED in each frame were converted into a form suitable for our animation program to read and our network simulation program to use. The data were converted from 3D to 2D by omitting the depth coordinate, which changed negligibly during the trials. The quality of the animation was considerably better than had been the case in the data sets used in the

[1] One being the author.

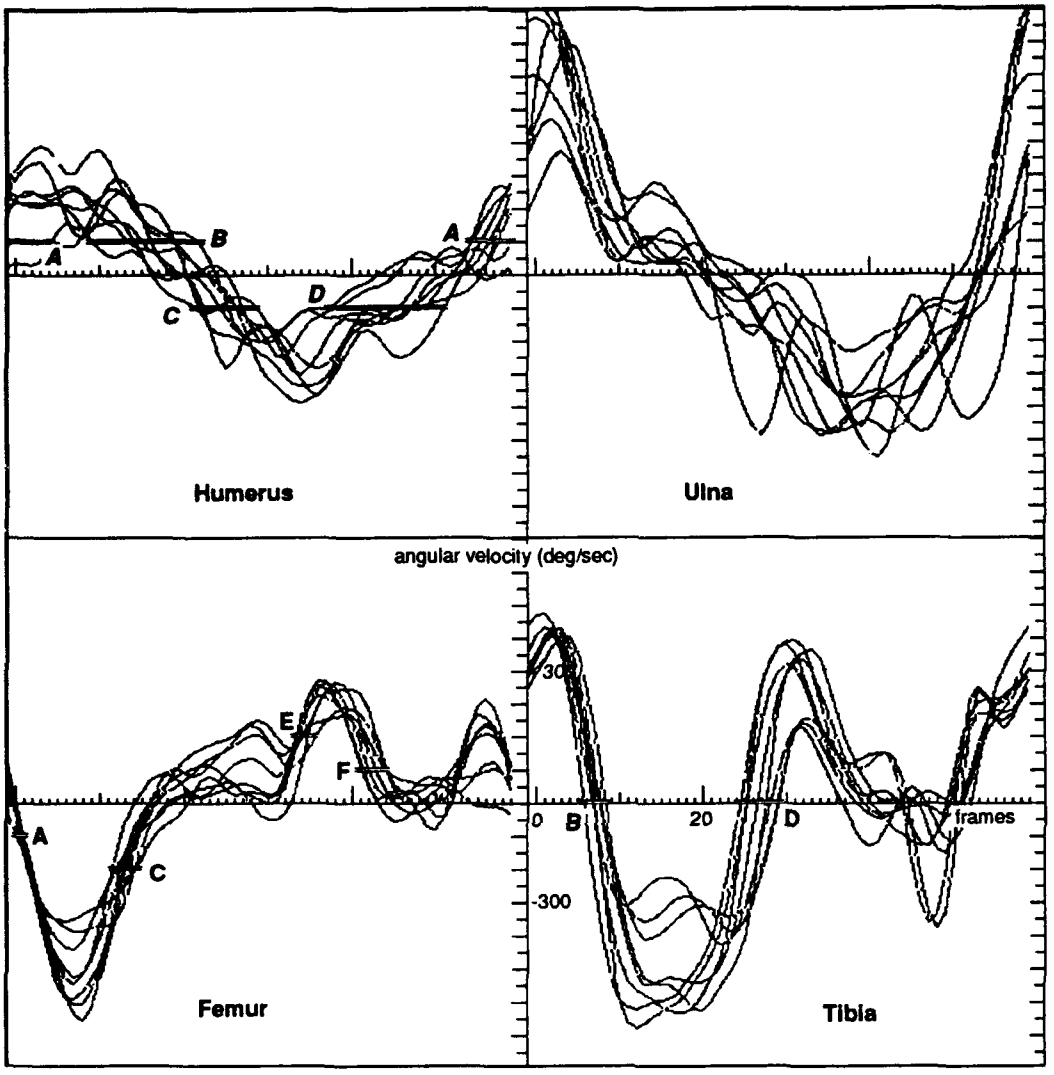


Figure 6.3: Angular Velocity of Limb Segments for Skipping Data

pilot experiments, there being almost no jitter. Initially we tried the architecture, features and scenarios from the first set of experiments. The result was dismal failure to discriminate any of the new data. It was quickly apparent that the features and scenarios used in the first set of experiments were much too specific to the particular data samples (a single sample of each gait) which were available for those experiments.

We constructed new features and scenarios, using the same principles as before. First, the raw data were converted into orientation angles for each of the limb segments. Software was written to graphically display and edit this data in order to smooth it and ensure that the starting and ending point of a cycle were identical (as in the first set of experiments, so that multiple cycles and arbitrary initial phase could be presented). Finally, the data were resampled to produce 60 frames per cycle for each set of data, thus normalizing out any

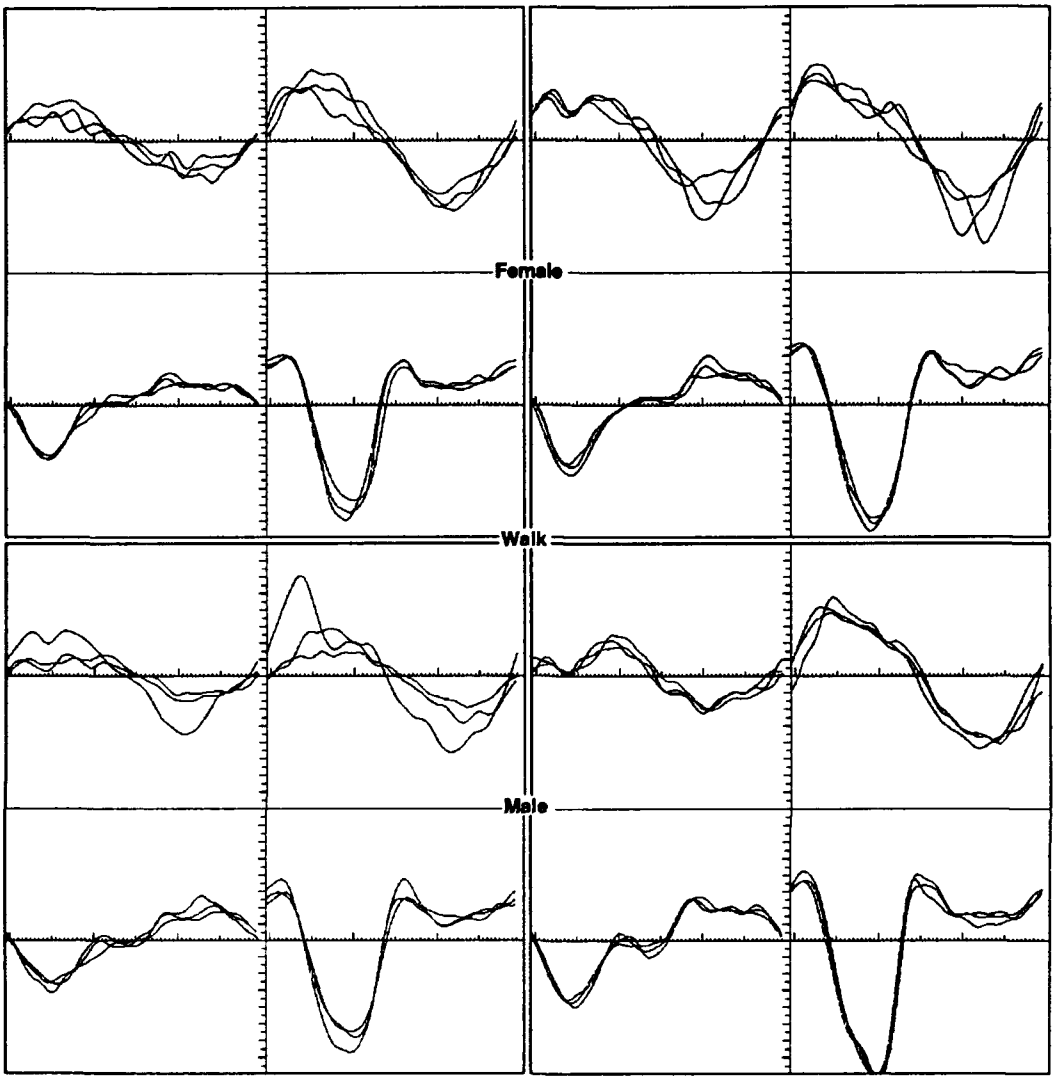


Figure 6.4: Walking Data per Individual Actor

differences of cycle time. As in the first set of experiments, distal data were generated by duplicating the proximal data and phase shifting 180°. We developed new features and scenarios for this data, but discrimination was still not reliable. Upon further examination of the data, it became apparent why this was the case.

### Experiments with Actor Dependent Scenarios

Figure 6.1 is a screeendump from the graphical editor, showing the angular velocity of each of the proximal limb segments over time[2] for walking. Figures 6.2 and 6.3 show

[2]humerus - upper arm; ulna - forearm; femur - thigh; tibia - calf.


Figure 6.5: Running Data per Individual Actor

the corresponding graphs for running and skipping. For each of the gaits there is a clear general pattern, but a significant amount of variation between individual samples. When the samples are grouped by actor, the graphs become much cleaner. Figure 6.4 is a screeendump showing the walking data plotted for each of the four individuals. The upper set of graphs show the samples from the female actors, the lower set from the male actors. Figures 6.5 and 6.6 show the corresponding graphs for running and skipping (recall that one of the males was unable to skip, and there were only two good skipping samples for the other male). Variation of samples within an individual is much smaller than variation between individuals, particularly for the leg segments (femur and tibia). There is an obvious analogy with speech recognition here. In that field, it has long been recognized that speaker-independent speech recognition is much harder than speaker-

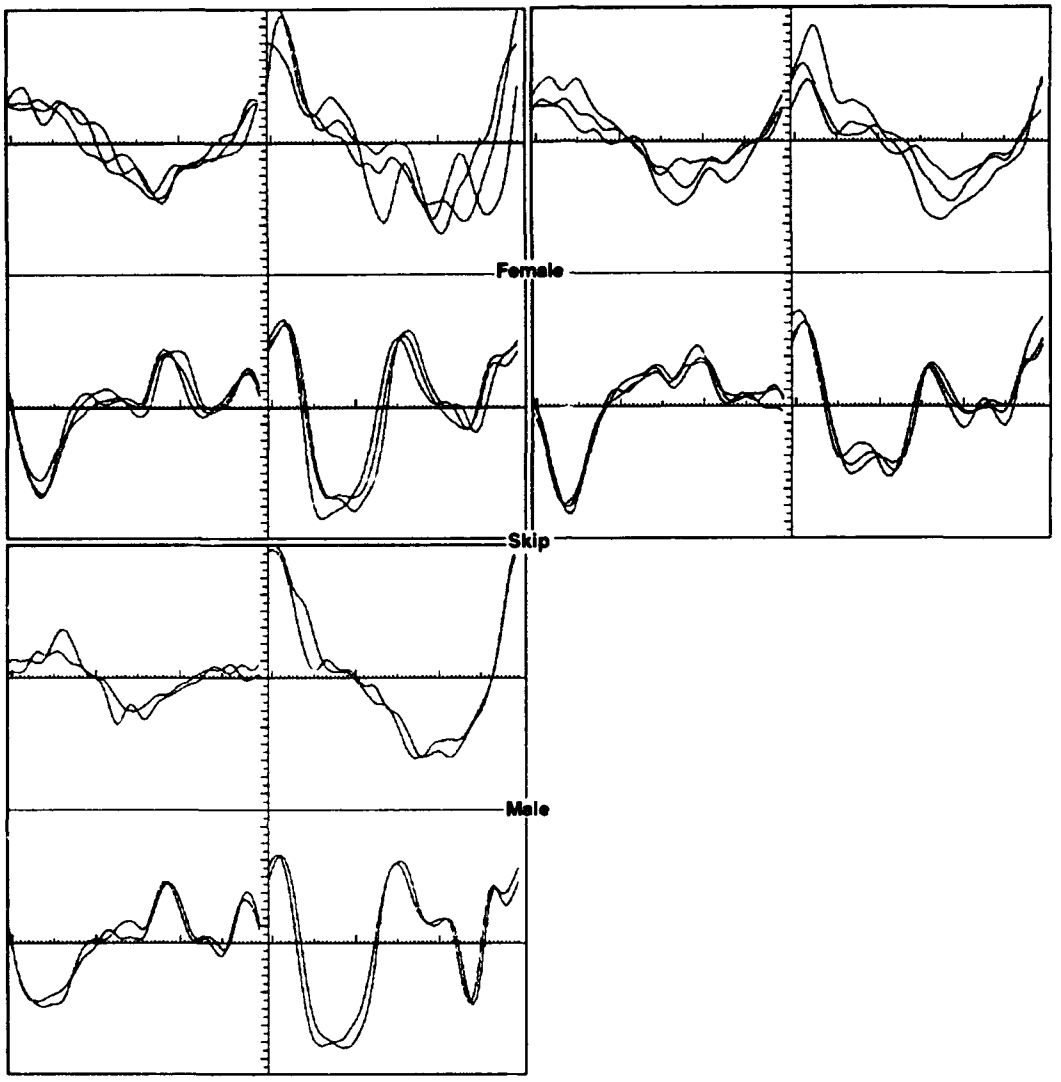


Figure 6.6: Skipping Data per Individual Actor

dependent recognition, largely because of the difficulty of constructing adequate speaker-independent phoneme models. Inasmuch as gait production and speech production are both motor skills, one would expect different individuals to exhibit different characteristics. Perhaps it is because of the characteristic graphs illustrated in Figure 6.4 that recognition of sex or identity of an individual from their gait is possible [Cutting and Kozlowski, 1977; Kozlowski and Cutting, 1977].

Accordingly, we ran a preliminary set of experiments using actor-dependent scenarios. There were three samples of each gait for each individual, so there was still a range of data to be accounted for by each scenario. We judged discrimination by the ability of the network to achieve a high-level of activation in the target scenario, and to keep activation in scenarios representing different *gaits* low. Scenarios representing the target gait but for a

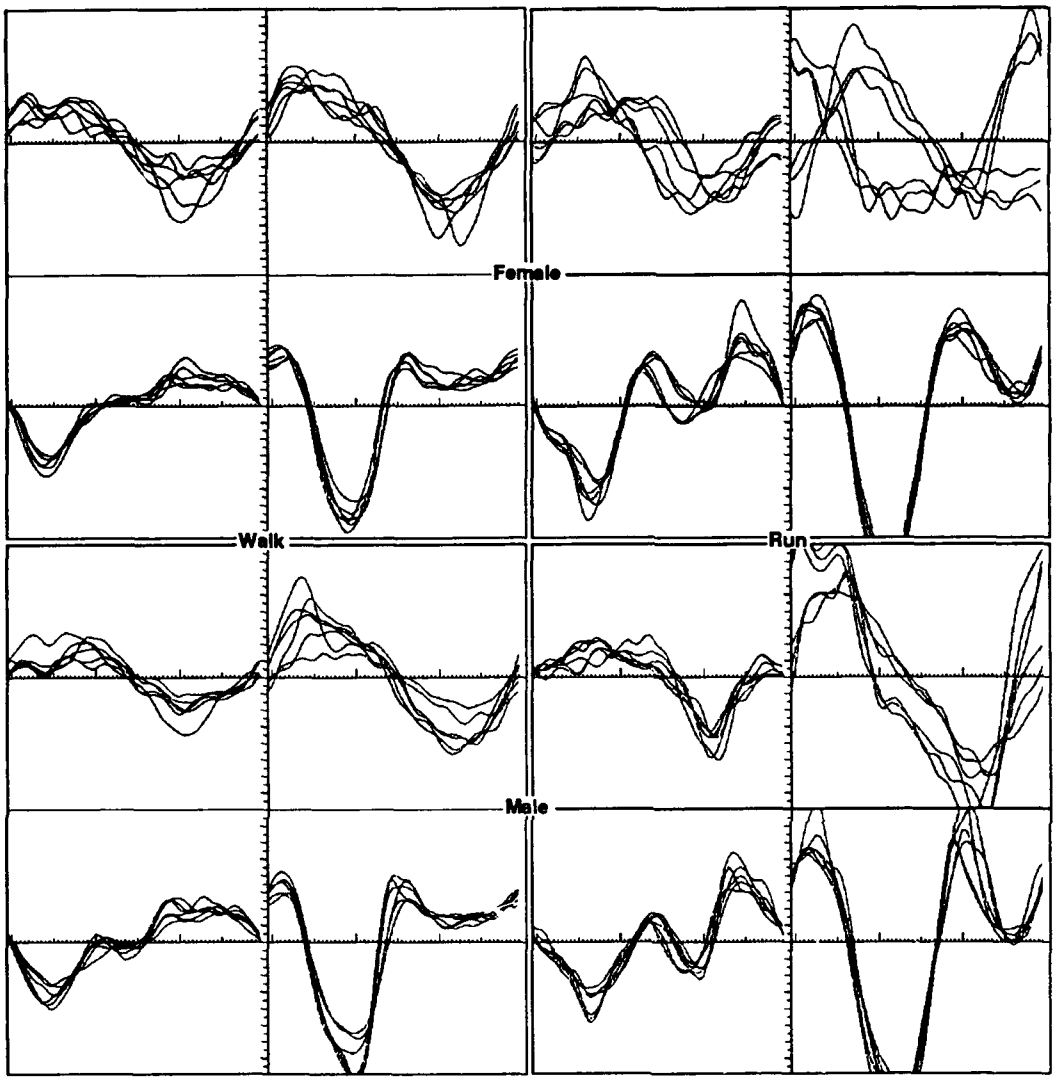


Figure 6.7: Walking and Running Data Grouped by Gender

non-target individual were allowed to become active. Experiments along the lines detailed in later sections of this chapter were carried out. Bugs in the program and better parameter settings were identified and the appropriate changes made. The results were on the whole positive, but we do not include them here because they are superseded by the experiments reported below. In a second set of experiments we used gender-dependent scenarios. The data samples from the two males were grouped and scenarios derived from them; likewise for the females. The graphical plots of the data grouped by gender for walking and running is shown in Figure 6.7. Once again programming bugs were found and fixed, and as a result of these experiments the spatial constraints in scenario composition were added. Some of the previous set of experiments were re-run using the gender-dependent scenarios. Again, the results were generally positive but are not reported because the final set of experiments,

using actor-independent scenarios, was successful.

**Actor-Independent Scenarios**

As a result of the improvements made with the actor- and gender-dependent scenarios, we ran the experiments again with actor-independent scenarios, i.e., using six scenarios (one for each of the three gaits, leftward and rightward) to handle all 32 gait samples. This time discrimination was successfully achieved in all cases. The remainder of this chapter describes in more detail the experiments on actor-independent gait recognition.

## 6.3 Feature and Scenario Design

We were unable to use exactly the same design rules as those used in the experiments reported in Chapter 5 for several reasons. In the scenarios, the interval units had to be more flexible. Instead of representing a single time interval between events, we needed to represent a range of time intervals in a single interval unit. In the feature hierarchies, we had to adjust the rules for generating new features to achieve a reasonable number of features at each level. These adjustments were necessary because the rules developed for the first set of data were much too specific to that limited set of data. In addition, we decided to omit the shape features altogether, in an effort to simplify the architecture for the purpose of achieving a better understanding of which aspects were truly critical.

### 6.3.1 Segment Level Features

We dispensed with shape features. The motion features were similar to those employed in the first set of experiments. Six angular velocities were coded for: ($\leq -200$, -200 to -100, -100 to 0, 0 to 100, 100 to 200, $\geq 200$ degrees/sec). Segment orientation was coded for by quadrant (every 90°). This resulted in 24 different features at the segment level.

### 6.3.2 Component Level Features

Component level features were very similar to those used in the first set of experiments. The only difference was that only one set of concentric circles was used to create remote regions (see Figure 5.2). The data sets required 115 component-level features.

### 6.3.3 Assembly Level Features

Assembly level features posed something of a problem. Given 115 component-level features, there are 6670 different possible pairs of component-level features. The data sets

utilized 1086 of these pairs. Although the simple thing to do is to have one assembly-level feature for each pair, this was considered too many either from a practical or a theoretical viewpoint, since the features are duplicated at every assembly-level location. The simple practical solution was to designate two separate assembly level features for each scenario event, one specialized for enabling conditions and the other for support. These event-specific features could then be activated by any one of the combinations of component-level features that corresponded to the enabling or support of the event. This solution is unrealistic for a large-scale vision system - just as one cannot have a separate model of walking for each individual, one cannot have separate assembly-level features for each event in each scenario. However, this was the solution adopted, resulting in 120 different assembly-level features (3 actor-independent gaits and 2 directions - leftward and rightward - for each gait, with 6 leg-scenario events and 4 arm-scenario events, and two features per event). There are two reasons why this was the right thing to do. First of all, it was simple and allowed us to validate the architecture. Second, the same effect could be achieved by having a distributed representation over a smaller set of units. Gradient descent learning algorithms are good at developing efficient distributed representations, but that is not the focus of our research. Therefore we used event-specific assembly-level features. As a consequence, the P-binding mechanism became superfluous: it was designed to mediate competition between scenarios for activation of assembly-level features, and in this case, because the features were event specific, there was no competition. However, the attention mechanism based on competitive proxy units compensated for the lack of P-binding.

### 6.3.4 Assembly Level Scenarios

Rather than attempt to automate the process of discovering useful event boundaries, as we had done for the first set of experiments, we decided to manually specify events for the wider range of data in the second set. The events were chosen by examining the angular velocity graphs (e.g., Figure 6.1) and looking for points where the segment-level motion feature boundaries occurred at roughly the same time in each data sample. Recall that the motion boundaries coded in the segment-level units were at $\pm 0$, $\pm 100$ and $\pm 200$ degrees/second. Figure 6.1 marks the crossing points selected for walking events. Six events (A - F) were selected for each leg-scenario and four (*A - E*)for each arm-scenario. Figures 6.2 and 6.3 show the events selected for running and skipping. The arm-scenarios were developed later in the series of experiments, and less attention was paid to finding events that would discriminate gait than for the leg events. For the leg-scenario events, care was taken to try to minimize the variability in time interval between events, because time intervals were designed into the scenarios as one of the important factors for discrimination.

Given these enabling conditions for the events, the time intervals were determined for each gait sample. From these time intervals, the plateau range for each interval unit (see Figure 3.6) was determined. Appendix A specifies the scenarios that resulted from this analysis.

### 6.3.5 Object Level Scenarios

Object level scenarios were implemented in a manner similar to that described in the previous chapter. As before, events from the constituent assembly-level scenarios were grouped together to form single object-level events. The assembly-level event with the lowest variability in its timing was linked to the object-level event with a delay link to provide the enabling input to the event. The other assembly-level events in the group could have been linked to the object-level event via an interval unit to provide supporting activation to the event, but in these experiments they did not contribute to the object-level scenarios. The distinction is important. The input that arrives along a delay link contains a spike of activation from the onset of the assembly-level event. This spike is the enabling condition that causes the object-level event to spike. Given the fact that there is uncertainty in the timing of particular assembly-level events, the best approach is to use the one with the least uncertainty as the basis for the object-level event onset.

In addition, the spatial constraint between arms and legs was added to the scenarios (i.e., arms must be approximately above legs). This was done by linking the assembly-level proxy units for a gait to the object-level summator unit for that gait. The summator unit then checked that proxies for arms and legs were active and in the appropriate spatial relation (links from proxies were labeled by spatial position). A further constraint to encourage consistency between the object- and assembly-level scenarios was added. This constraint was that the activation of the assembly-level summators for a gait should tend toward the activation of the object-level summator for that gait. It was implemented with a link from the object-level summator to the assembly-level summators.

## 6.4 Network Simulation Details

Initially we implemented the motion feature hierarchy and the assembly-level leg scenarios, under the assumption that most of the discriminating power derives from motion information in the leg movements. O- and P-binding were turned off. Initially the proxy based attention mechanism (the what/where attention map) was also turned off. Each gait cycle consisted of 60 frames, which represented approximately 1 second of real-time data (less for running, more for skipping). The event units integrated assembly-level features over patches consisting of three overlapping assembly-level locations, instead of the four overlapping locations used in the first set of experiments (see Figure 4.8 for an illustration and accompanying text for an explanation). This was done to reduce spatial crosstalk between the leg and arm input locations. As before, the simulation step was set at 1/300 second real-time equivalent, giving 5 simulation steps per frame. We performed some initial experiments not reported here, which indicated that leftward- and rightward-movement scenarios were not confused with each other by the network. Therefore, in the following experiments we presented only rightward-movement scenarios, e.g., walking-right (although the scenario hierarchy represented both leftward and rightward movements).

| *network* | *scenarios* | | *features* | | *proxies* | | *Total* | |
|---|---|---|---|---|---|---|---|---|
| | *units* | *links* | *units* | *links* | *units* | *links* | *units* | *links* |
| *walk* | 96 | 342 | 1428 | 33685 | 56 | 3080 | 1580 | 37107 |
| *walk/run* | 184 | 658 | 2729 | 78947 | 112 | 6608 | 3025 | 86213 |
| *walk/run/skip* | 280 | 1000 | 3367 | 119212 | 168 | 10584 | 3815 | 130796 |
| *new gait* | 100 | 350 | 1000 | 40000 | 56 | 4000 | 1156 | 45000 |
| *full network* | 280 | 1000 | 13020 | 461295 | 228 | 14364 | 13528 | 476659 |

Table 6.1: Hardware required for different system modules

### 6.4.1 Connectionist Hardware Required

The numbers of units and links required to simulate one cycle of a single sample of walking are shown in Table 6.1. The numbers for other samples are comparable. The table shows the number of units and links required for each system module - the scenario hierarchy, the feature hierarchy, and the attention map. The numbers of links for the features and proxies include the links used to connect those units to the scenarios and to each other. On the first line we show the numbers required to model the walking gait alone. On the second line we show the numbers for walking and running. The third line shows the numbers for all three gaits. The table shows that most of the hardware was required for the feature hierarchies and connections between the feature hierarchy and scenario hierarchy. Three gait models required only 280 units and 1000 links to implement. We estimate the numbers of additional units and links for a new gait on the fourth line of the Table. These numbers overestimate the effect of adding a new gait to a system with many gaits, for which only a small number of new features would be needed.

In the feature hierarchies, we implemented only those input locations where there was input, and those component and assembly locations receiving activation from those input locations. For this gait sample, we implemented 8 input locations, 13 component locations and 14 assembly locations. Each input location required 24 feature units, each component location 115 feature units and each assembly location 120 feature units. Thus we needed 3367 feature units for all three gaits. For the attention mechanism, we needed 12 proxy units at each assembly-level location (3 gaits, 2 orientations, 2 assembly-level scenarios per gait per orientation), requiring another 168 units. We can extrapolate from these numbers to estimate the number of units and links required to build the complete architecture for a visual area about the size covered by a human figure. For an 8x20 visual area, we need 160 input locations, 60 component locations, and 19 assembly locations. This would require 13020 feature units and 228 proxy units for all three gaits. We estimate the number of links as shown on the final line of Table 6.1.

## 6.5 Experiments

The experiments were conducted in sequence, starting with the simplest version of the architecture, and turning on mechanisms as the results of prior experiments indicated. This showed precisely when each mechanism was needed. Finally, experiments were performed to test sensitivity to other capabilities of the system. In the following discussions, we use two important parameters for judging performance:

- **threshold** - the minimum activation a scenario must achieve and maintain to be considered a candidate for the winner.
- **separation** - a value for the ratio of the most active non-target's activation to the target scenario's activation. Separation is achieved once the ratio reaches this value and never falls below it again.

### 6.5.1 Introduction

The **basic** architecture included the motion feature hierarchy and assembly- and object-level scenarios. It omitted the what/where attention map (the proxies), the top-down feedback from object-level to assembly-level scenarios, the spatial constraint between arm and leg proxies, and the constraint tending to equilibrate the object and assembly-level scenarios. The improvements that were found to be necessary were:

1. **Attention Map**. This incorporated the attention map, with three components: 1) Using the output of each proxy unit to increase the significance of particular features at its location to particular events in its scenario. This is the basic bottom-up component in the attention mechanism. 2) Using simultaneous proxy and interval unit activation to enhance features that are expected to occur imminently. This is the top-down component of the attention mechanism. 3) Mutually inconsistent proxies inhibit each other, and the proxies are given a "resting level" of activation. Activation levels below resting level result in inhibition in the attention mechanism, those above result in facilitation.
2. **Compositional Constraints**. This added the spatial constraint between arms and legs, and the tendency for assembly-level summators to have the same level of activation as their gait's object-level summator.
3. **Synchronization Constraint**. This added the top-down feedback from object-level events to assembly-level events. This feedback tended to synchronize the resonance in the object- and assembly-level scenarios.

Figure 6.8 illustrates the effect of each mechanism for a single input sequence (a particular gait sample, visual location and initial phase). It shows qualitatively the gain in

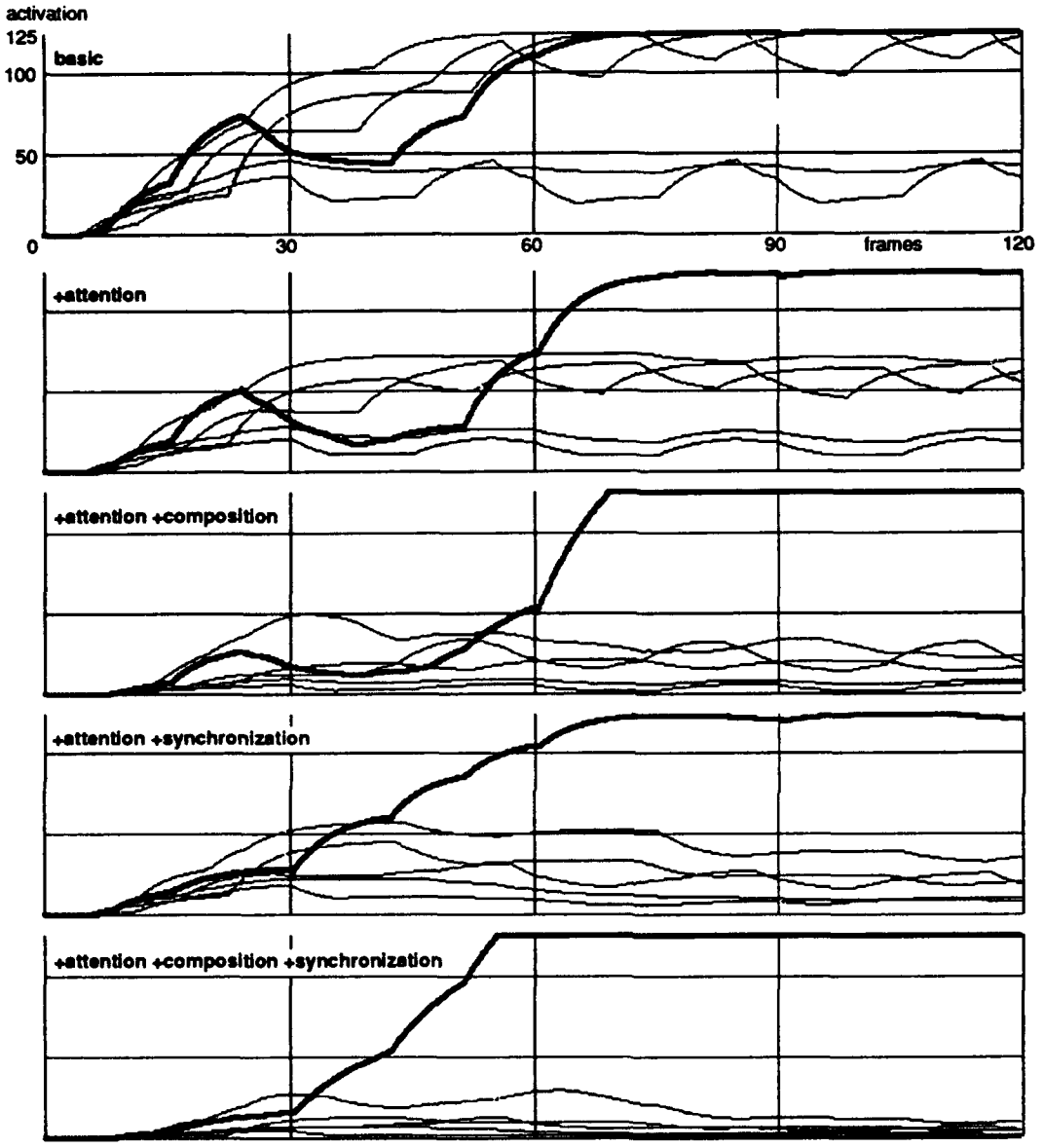


Figure 6.8: Result as Each Mechanism is Added

recognition power that each mechanism makes. In the experiments described below, we report results as each mechanism is added for a comprehensive set of trials. In Figure 6.8, the traces show performance for systems with different mechanisms, from top to bottom:

1. **Basic.** Recall that there are six modeled gaits (walk-left, run-left, skip-left, walk-right, run-right and skip-right). The thick line shows the activation of the target scenario, that is the scenario representing the gait present in the input sample. The five thin lines show the activation of non-target gaits. For this input sequence, the basic architecture was unable to discriminate the gait, three of the non-target scenarios

also reaching high levels of activation

2. **Attentive**. The second trace shows the performance, for this gait sample, of the attentive system, which consisted of the basic system with the attention mechanism added. Adding the attention mechanism to the basic architecture resulted in significant asymptotic suppression of the non-targets, but only after about 60 frames (1 second) of input presentation.

3. **Composed**. The third trace shows the performance of the composed system, which consisted of the attentive system with the compositional constraints added. This enhanced activation of the target scenario and caused further asymptotic suppression of the non-targets.

4. **Synchronized**. The fourth trace shows the performance of the synchronized system, which consisted of the attentive system with the synchronization constraint added. This improved the speed of response (separation of the target from the non-targets occurs around frame 45 instead of frame 60), and also increased asymptotic suppression of the the non-targets.

5. **Full**. The fifth trace shows the performance of the full system, which consisted of the attentive system with the compositional constraints and the synchronization constraint added. The result was early separation of the target from the non-targets (around frame 35), rapid achievement of full activation in the target, and a high level of suppression of the non-targets.

### 6.5.2 Experiment 1: Comparing Architectures

In this experiment we investigated how well the different architectures introduced above could discriminate across all gait samples and topologically distinct input locations. We tested all 9 topologically distinct locations, starting the input sample at a fixed initial frame (sensitivity to initial phase was tested in a later experiment). There were 32 different input sequences (3 gaits x 4 actors x 3 samples of each, less one skipping sample for one actor and all the skipping samples for the actor who could not skip). Thus we had 288 trials. In each trial we presented the sequence for 2 cycles (120 frames or 600 simulation steps). Each trial required approximately 25-50 minutes CPU time on a 16 MIPS Sun 4/40, the times varying depending upon which mechanisms were used. We ran the simulations overnight in parallel on a network of 20 such machines, for several nights. The result of each trial was a file of data from which simulation traces such as those shown in Figure 6.8 could be plotted. The 288 trials produced approximately 4 megabytes (compressed) of data. This experiment was repeated with each of the five architectures outlined above, culminating with the full architecture. In all cases, the activation levels were recorded from the object-level summator unit for each gait.

We set our criteria for successful discrimination to be:

| | |
|---|---|
| threshold | 80 |
| separation | 0.40 |

This means the target scenario must have reached and maintained an activation level of 80 (maximum activation was 125), and no non-target could achieve a level of more than 40% below that of the target. If these criteria were first satisfied, for a given input sequence, at frame $n$, then we say discrimination of that input sequence was achieved at frame $n$.

#### Deficiencies Revealed in the Basic System

The basic architecture achieved the discrimination threshold level of activation in 99% of the trials by frame 90, but it performed very poorly in separating out the target scenario from the non-targets, reaching separation in only 1% of the trials by frame 90. If the requirement for separation was reduced to 10% (i.e., non-targets had to stay at least 10% below the level of the target), then 22% of the trials achieved separation.

In summary, few of the input sequences were correctly discriminated using the basic architecture. After examining the processing that occurred, four main problems were identified:

- **Spatial Crosstalk**. The arm movements were confused transiently with leg movements. Additionally, the forearm and thigh segments, which transiently were close to each other, activated leg features.
- **Poor resonance**. In a properly activated scenario, there are two waves of activation, 180° out of phase, flowing around the scenario network, corresponding to the two

| *system* | *frames presented* | | | | | |
|---|---|---|---|---|---|---|
| | 30 | 45 | 60 | 75 | 90 | 105 |
| Basic | 0 | 1 | 1 | 1 | 1 | 4 |
| Attentive | 7 | 46 | 73 | 80 | 86 | 89 |
| Composed | 12 | 87 | 94 | 97 | 97 | 97 |
| Synchronized | 4 | 52 | 91 | 98 | 99 | 100 |
| Full | 2 | 86 | 100 | | | |

Table 6.2: % of Samples Correctly Discriminated - Different Systems

interpretations of the leg/arm pairs as proximal and distal. For some input sequences, one of these waves took a long time to establish itself (because of sensitivity to the initial phase). The traces in Figure 6.8 are from such a sequence.

- **Slow response.** Often the target scenario took longer than we would have liked to establish itself. We took one cycle (two steps of the gait) or 60 frames as a figure to aim for.

- **Poor separation.** In most instances, the network failed to achieve adequate separation of the target-scenario from the non-targets.

The first two problems are problems of competence. The basic architecture is inherently prone to spatial crosstalk problems; and the scenarios are not guaranteed to be able to identify the correct time alignment with the input. The second two problems are performance problems. The target scenario did achieve a more consistently high level of activation than the non-targets, but insufficient or intermittent separation. In the few cases where the criteria were met, achieving discrimination took longer than wanted.

### Results for Enhanced Systems

In Tables 6.2-6.4 we summarize the results for the five different architectures which were introduced above. Table 6.2 shows the overall performance of each architecture given the criteria of threshold activation and separation described above. Table 6.3 shows how well each architecture does in achieving the threshold level of activation in the target scenario. Table 6.4 shows how well each architecture does in achieving the 40% level of separation between the target scenario and the non-targets.

Examination of these tables shows that the function of the extra mechanisms which were added to the basic architecture is primarily to improve the separation between the target and non-target scenarios. Table 6.3 shows that even the basic architecture achieved the threshold level of activation in the target scenario by frame 60 in 97% of the trials. But Table 6.4 shows that the basic architecture achieved adequate separation of the target

| *system* | *frames presented* | | | | | |
|---|---|---|---|---|---|---|
| | 30 | 45 | 60 | 75 | 90 | 105 |
| Basic | 42 | 89 | 97 | 98 | 99 | 99 |
| Attentive | 9 | 77 | 94 | 99 | 99 | 99 |
| Composed | 12 | 91 | 99 | 99 | 99 | 99 |
| Synchronized | 4 | 63 | 97 | 100 | | |
| Full | 2 | 86 | 100 | | | |

Table 6.3: % of Samples Correctly Discriminated - Threshold

| *system* | *frames presented* | | | | | |
|---|---|---|---|---|---|---|
| | 30 | 45 | 60 | 75 | 90 | 105 |
| Basic | 0 | 1 | 1 | 1 | 1 | 4 |
| Attentive | 16 | 49 | 73 | 80 | 86 | 89 |
| Composed | 67 | 91 | 94 | 97 | 97 | 97 |
| Synchronized | 18 | 68 | 92 | 98 | 99 | 100 |
| Full | 76 | 98 | 100 | | | |

Table 6.4: % of Samples Correctly Discriminated - Separation

from the non-targets in almost no trials. The attention mechanism performs the bulk of the work in achieving separation - in almost 90% of the trials separation is achieved in the attentive system (Table 6.4). The composition constraints increase the percentage of trials in which separation is achieved, and decreases the time taken to reach threshold activation and separation. This is because the composition constraints suppress some non-targets, increasing both the chance for the target to achieve separation and the speed with which the competition for dominance is settled. The synchronization constraint is somewhat more powerful in increasing the rate of separation, but not as powerful in increasing the speed with which separation is achieved. The combination of all three mechanisms is required to achieve fast, correct discrimination.

### Dynamics of the Attention Map

The mechanisms that achieve activation of the target scenario constitute the "What" process in this system. The attention map subsystem is the "Where" process. The implementation of the two processes were designed to aid each other, with the expectation that they would cooperate in solving the problem of gait recognition. Figure 6.9 illustrates the solution to the "Where" problem that is found by the attention map for a sample of walking. The five

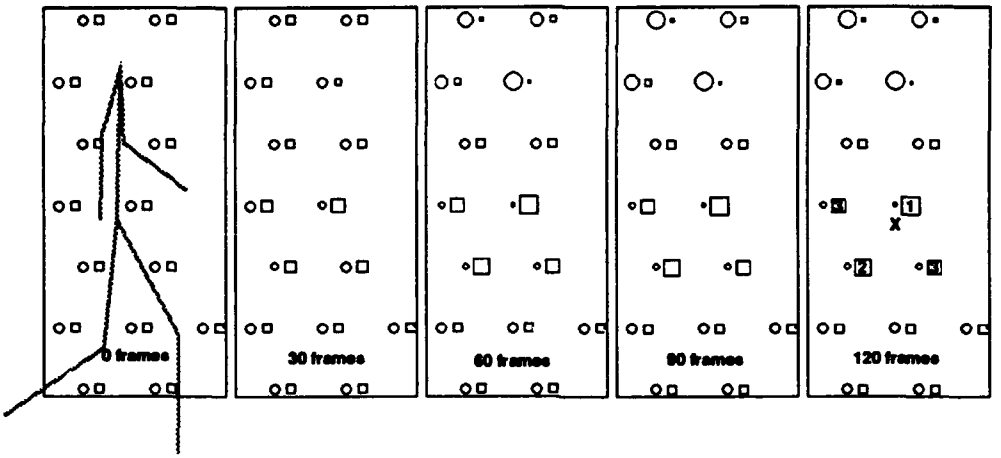


Figure 6.9: Evolution of Attention Map over time

panels show the activation of some of the attention map units after 0, 30, 60, 90 and 120 frames of input. Each panel shows two units, one circular and one square, at each location in the map. The input biped is overlaid on the first panel to indicate the scale and resolution of location information. The square units represent the proxies for the "legs-walk-right" scenario; the circular units the "arms-walk-right" scenario. Initially the attention map is neutral: all proxy units are at resting activation (activation level is indicated by the size of the icon). By frame 30, the square units at the legs location (the hip) are becoming more active, and the circular units at that location less active. By frame 60, the square units are dominant at the legs locations, and the circular units are dominant at the arms location (the shoulder). Over the next 60 frames there is little change. The "Where" process has settled on the solution by frame 60. Notice that location information is coarse-coded amongst up to four proxy units. In the final panel of the Figure, the location of the legs is marked with an "X", and the closest, second closest and third closest locations are marked 1, 2, and 3 respectively.

It is of interest to compare the dynamics of the "What" and "Where" processes. Figure 6.10 shows traces for the "What" and "Where" processes for same gait sample used above. The upper plot shows the activation of the leg-movement scenarios, with the target (legs-walk-right) shown by the heavy line and the non-targets by the thin lines. The second plot down shows the activation of the proxy units for "legs-walk-right" at the different spatial locations, with the target location (i.e., the proxy location closest to where the legs were) shown by the heavy line and the non-target locations shown by thin lines.

The first thing to notice here is that the evolution of the "What" and "Where" processes, as indicated by the heavy lines, parallel each other closely. The target scenario trace begins to separate from the non-targets around frame 15, at the same frame as the target proxy starts to separate itself from the non-targets. The two processes settle into solutions in parallel, the "Where" process not needing the solution to the "What" process before it can begin to settle.

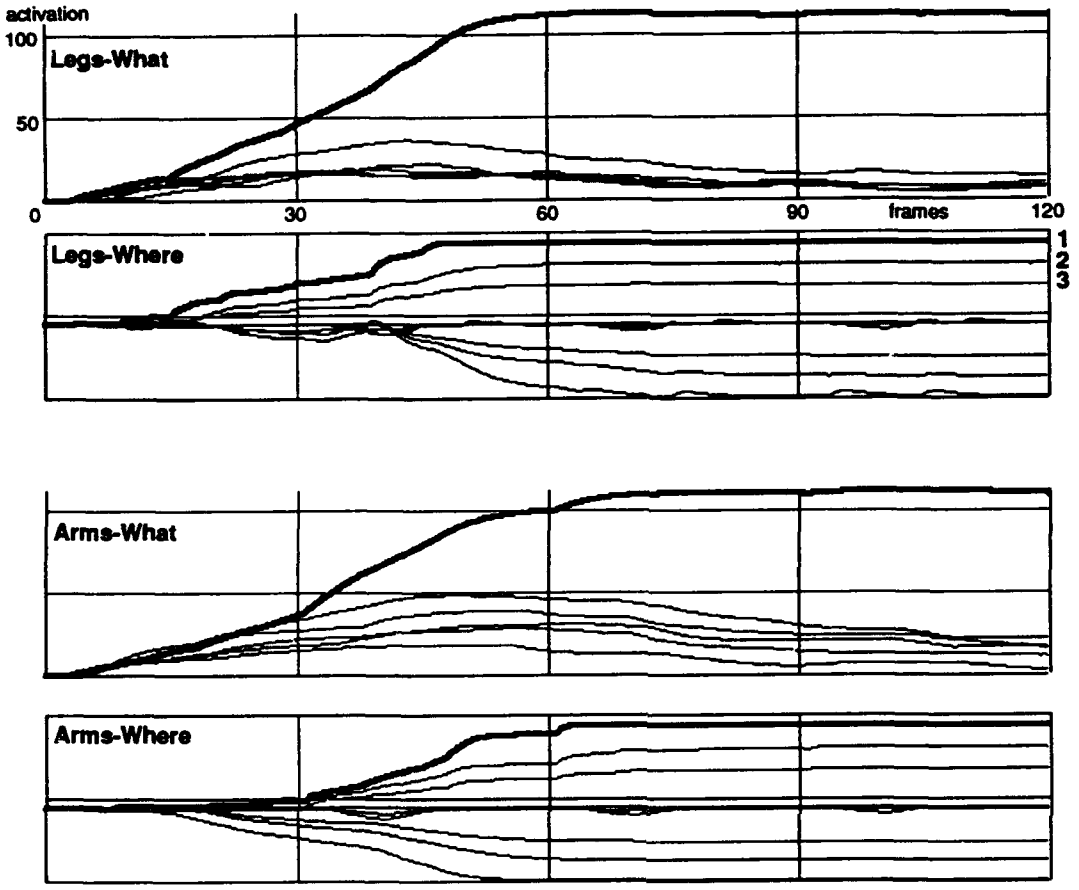


Figure 6.10: Dynamics of "What" and "Where" processes

The second thing of note is that in the trace of the proxy units, some of the non-targets have fairly high levels of activation. This is because of the coarse coding of location information. The traces marked 1, 2 and 3 correspond to the proxies identically marked in Figure 6.9. The non-target traces which asymptote at an activation of 45 are from proxies at the resting level of activation. The traces which asymptote below 45 are from proxies near the location of the arms, which are suppressed by the proxies at those locations selected by arms-walk-right.

The lower pair of plots shows the evolution of activation in the arms scenarios and proxies. Again the parallel settling to solutions and the coarse coding of location is apparent.

| *experiment* | *frames presented* | | | | | |
|---|---|---|---|---|---|---|
| | 30 | 45 | 60 | 75 | 90 | 105 |
| All phases | 9 | 91 | 100 | | | |
| Single phase | 2 | 86 | 100 | | | |

Table 6.5: Initial Phase: % of Samples Correctly Discriminated

### 6.5.3 Experiment 2: Initial Phase

In this experiment we ran a set of simulations to investigate sensitivity to initial phase using the full architecture. A single input example of each individual executing each gait was presented at a single location, with initial phase incremented by 12° per trial. For wireframe figures there is symmetry about the 180° point in the cycle, so that 12° increments give 15 distinct initial phases. This resulted in 165 trials (11 gait samples x 15 initial phases). The results are shown in Table 6.5. As can be seen, there is no significant sensitivity to initial phase.

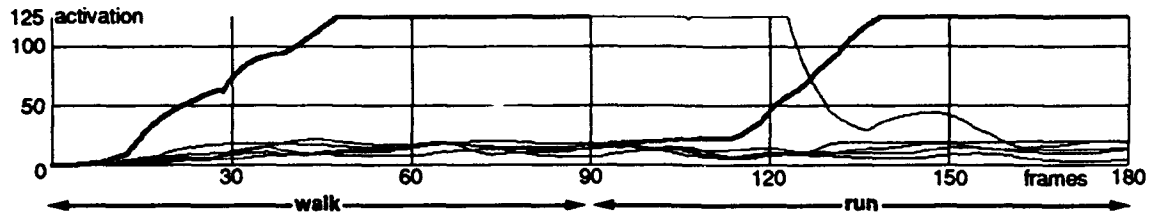


Figure 6.11: Transition from Walking to Running

### 6.5.4 Experiment 3: Transition to Another Gait

We conducted one experiment to check the transition from one gait to another. For one of the actors, we selected at random one walking and one running gait sample. We found a frame in the walking sample that was very similar to one in the running sample. We presented the walking sample for 90 frames to establish activation in the walking scenario. Then, without resetting the network, we presented 90 frames from the running scenario. We arranged that the terminating walking frame and the commencing running frame were similar. Figure 6.11 shows the results. Initially the walk-right scenario dominates (the thick line which dominates from frame 45 to frame 90). The transition in the input from walking to running occurs at frame 90. After frame 90, the thick line represents the new target, the run-right scenario. After a delay of approximately 25 frames, the run-right scenario begins to gain activation (the thick line). 45 frames after the transition in the input, at frame 135, the system has switched its interpretation from the initial scenario (walk-right) to the final scenario (run-right).

| *experiment* | *frames presented* | | | | | |
|---|---|---|---|---|---|---|
| | 30 | 45 | 60 | 75 | 90 | 105 |
| Occlusion | 3 | 78 | 99 | 100 | | |
| No occlusion | 2 | 86 | 100 | | | |

Table 6.6: Occlusion: % of Samples Correctly Discriminated

### 6.5.5 Experiment 4: Occlusion

In this experiment we simulated the self-occlusion present in Moving Light Displays made from real human movement. During normal gait parallel to the image plane, the distal joints are occluded by the body or by the proximal limbs intermittently. The proximal hip is itself occluded twice in each gait cycle by the proximal arm. We simulated this by turning off the input units for some of limb segments. The rule used was that if a joint was occluded, then the units representing both limb segments connected to that joint were turned off (in the case of the distal wrist, this applies to the distal forearm only). The results are shown in Table 6.6. There is a slight overall slowdown in speed of discrimination compared to the no-occlusion (wireframe) case, but discrimination is achieved by frame 75 in all cases.

| *experiment* | *frames presented* | | | | | |
|---|---|---|---|---|---|---|
| | 30 | 45 | 60 | 75 | 90 | 105 |
| Clutter | 3 | 66 | 81 | 94 | 100 | |
| No clutter | 2 | 86 | 100 | | | |

Table 6.7: Comparison of results with and without clutter

### 6.5.6 Experiment 5: Clutter

In this experiment, background clutter was introduced together with a gait sample. The clutter occupied the same visual area as the sample. It consisted of 12 rods each rotating about its center, placed at random locations. The angular velocities of the rods were randomly distributed in the same range as the angular velocities of the limb segments in the sample. This resulted in 24 extraneous input units being active in each input frame (compared with 8 active input units required to represent the sample). In this experiment each of the 32 gait samples were presented, with clutter, at a particular location. The results generally showed that time-to-discrimination was greater but discrimination was achieved in every case. The comparison with the no-clutter results is shown in Table 6.7.

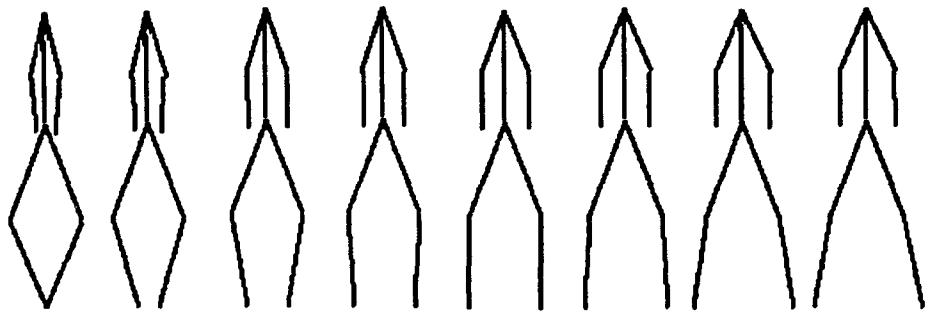

Figure 6.12: Nonsensical movement

| | *% of maximum activation* | | | | | | |
|---|---|---|---|---|---|---|---|
| | 10 | 20 | 30 | 40 | 50 | 60 | 70 |
| % trials | 79 | 13 | 4 | 2 | 1 | 1 | 0 |

Table 6.8: % of Trials in which any Scenario reaches a given level of activation

### 6.5.7 Experiment 6: Nonsense

In this experiment we presented a biped figure moving in a manner similar to a real gait sample, but sufficiently distinct that a valid model should reject the input as not matching any modeled gait. The movements were derived from each gait sample by replacing the motion of the distal leg and arm by the reflection of the proximal leg and arm about vertical axes passing through the hip and shoulder respectively. Figure 6.12 shows a sequence of frames from such a nonsensical movement. We generated these nonsensical movements for each gait sample, and presented them to the network at each of the 9 distinct input locations. These movements were a severe test of the network because each half of the input partially matches a gait scenario. The proximal-limb half of the input partially matches the scenario representing the gait sample from which it is derived. The mirror image half of the input partially matches the corresponding scenario for movement in the opposite direction (i.e., left instead of right or *vice versa*. In addition, the spatial structure of the input is very similar to the spatial structure of the modeled gaits. The results are shown in Table 6.8. This table shows the percentage of trials in which any scenario reached a given level of activation. As can be seen, a level of 30% of maximum activation was reached in only 4% of trials . Of the 250 trials, in only one did a scenario achieve over 50% of maximum activation, and this was only achieved transiently.

| *experiment* | *frames presented* | | | | | |
|---|---|---|---|---|---|---|
| | 30 | 45 | 60 | 75 | 90 | 105 |
| Two of two figures | 0 | 36 | 79 | 90 | 91 | 92 |
| One of two figures | 0 | 85 | 93 | 93 | 94 | 94 |
| Single figure | 2 | 86 | 100 | | | |

Table 6.9: % samples correctly discriminated for two spatially-separated figures

| *experiment* | *frames presented* | | | | | |
|---|---|---|---|---|---|---|
| | 30 | 45 | 60 | 75 | 90 | 105 |
| One of two figures | 0 | 0 | 0 | 0 | 0 | 0 |
| Single figure | 7 | 15 | 17 | 19 | 19 | 22 |

Table 6.10: % of samples correctly discriminated without attention

### 6.5.8 Experiment 7: Multiple Figures

In this experiment two figures executing different gaits were presented simultaneously. In the first condition, the figures were separated (one on the left of the visual field, one on the right). The results showed that in most cases both gaits were discriminated correctly, but time to discrimination was slightly longer. Table 6.9 shows the results. In the first row we show the percentage of samples in which both gaits were correctly discriminated. In the second row we show the percentage of samples in which at least one of the gaits was correctly discriminated. The third row gives the corresponding results for single figure presentation for comparison.

Parallel discrimination of two gaits was found to be due to the attention mechanism. When the attention mechanism was turned off, in no trial was either of the gaits successfully discriminated, even with the relaxed criterion of 10% separation. Table 6.10 shows a comparison of the results when attention is turned off in the single figure case with the results obtained in this experiment. The first row shows that without attention, no gait was discriminated when two gaits were presented simultaneously but in separated visual areas. In comparison, when a single gait is presented, 22% of the samples are correctly discriminated even with attention turned off.

In the second condition, the gait samples were presented at neighboring locations in the visual field (i.e., the figures occupied roughly the same visual area). Performance was found to be substantially worse than in the first condition. In a few trials both gaits were correctly discriminated, and in a majority of trials at least one of the gaits was discriminated. The results are shown in Table 6.11. The first row of the table shows percentage of trials in which both gaits were correctly discriminated. The second row shows the corresponding

| *experiment* | *frames presented* | | | | | |
|---|---|---|---|---|---|---|
| | 30 | 45 | 60 | 75 | 90 | 105 |
| Two of two figures overlapping | 0 | 8 | 10 | 13 | 15 | 16 |
| Two of two figures separated | 0 | 36 | 79 | 90 | 91 | 92 |
| One of two figures overlapping | 18 | 66 | 79 | 81 | 84 | 85 |
| One of two figures separated | 0 | 85 | 93 | 93 | 94 | 94 |

Table 6.11: % samples correctly discriminated for two figures (overlapping and separated)

numbers from the experiment with spatially separated figures. Performance is much worse with overlapping figures. We surmise that the attention mechanism tends to focus activation on one of the two competing scenarios. The third and fourth rows show the percentage of trials in which at least one of the gaits was correctly discriminated, in the overlapping and separated conditions respectively. Performance is worse when the figures overlap, but not substantially worse.

It is of interest to consider this second condition from the point of view of a noisy or cluttered input. If we designate, randomly, one of the presented gaits, as the target and the other presented gait as noise in the input, then these results show that the design exhibits some tolerance of background clutter that is closely related to the target (42.5% successful discrimination).

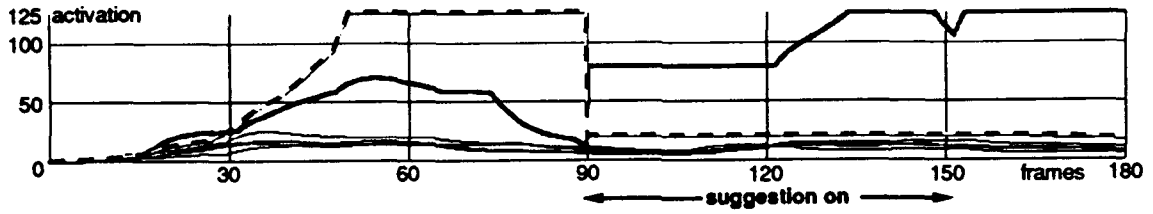


Figure 6.13: Perceptual Reversal due to Top-Down Suggestion

### 6.5.9 Experiment 8: Gestalt Reversal

In the final experiment, we presented two different gaits samples, walk-left and walk-right, simultaneously in roughly the same visual area. The intent was to see if we could induce perceptual reversal by injection of activation simulating top down "suggestion". These gait samples were selected by prior experimentation in which it was determined that the walk-right scenario dominated the walk-left scenario (symmetry being broken by the fact that the samples were not presented at exactly the same location). In the experiment, we presented 240 frames of both gaits. Figure 6.13 shows a trace of the the first 180 frames. The walk-right scenario trace is given by the dashed thick line, the walk-left scenario by the solid thick line. During the first 90 frames, the walk-right scenario establishes itself as the dominant interpretation, suppressing the walk-left scenario to a low level by frame 90. At this point we simulated top-down "suggestion" that the scene in fact contained walk-left rather than walk-right by suppressing the activity of the summation units of the walk-right scenarios (assembly and object level) and enhancing the activity of the summation units of the walk-left scenarios. This "suggestion" was applied for 60 frames of input. We suppressed the walk-right units to a level of 20 or below, and enhanced the walk-left units to a level of 80. After about 30 frames of this "suggestion", the walk-left scenario became sufficiently active that the enhancement had no further effect. After 45 frames of the "suggestion", the walk-left scenario had achieved maximum activation. After 60 frames of "suggestion", we turned it off. For the remaining 90 frames of the experiment, 30 frames of which are shown in Figure 6.13, the walk-left scenario remained at maximum activation (apart from the small dip around frame 150), completely dominating the walk-right scenario, which formerly had been dominant. This demonstrates perceptual reversal by top-down suggestion.

## 6.6 Summary

The experiments described in this chapter show that the processing mechanisms developed for the architecture are all essential in achieving fast, accurate discrimination of gait. The attention mechanism was needed to deal with problems of spatial crosstalk. Competition for attention was required to ensure adequate separation between the target and non-target scenarios. Higher level scenarios were needed to ensure rapid response through temporal synchronization and elimination of false positives through spatial composition. Given a single gait as input, the system made the correct discrimination in all cases within 60 frames of presentation, equivalent to one cycle of the gait. Examination of the dynamics of the attentive process ("Where") and the indexing process ("What") indicate that they function in a parallel, cooperative fashion. We showed that there is no significant sensitivity to initial phase of the input presentation, and that transition from one gait to another can be handled satisfactorily. The system was shown to be tolerant of the type of occlusion found in real images sequences of Moving Light Displays. Stimuli superficially similar in both spatial and temporal structure to the modeled gaits were rejected by the system. Background clutter consisting of rotating line segments slowed the discrimination process slightly, but discrimination was achieved in all cases. Background clutter consisting of a second gait sample overlaid on the target sample did disrupt discrimination significantly, but even in this case over 40% of the samples were correctly discriminated. In an experiment going beyond the design parameters for the system, we found that when two gait samples were presented in disparate spatial regions of the visual field, in over 90% of the trials both samples were correctly discriminated. A final experiment demonstrated perceptual reversal by applying top down suggestion into high levels of the network.

There are many more interesting experiments that could be run, for example: using new gait samples to test the generality of the scenario models; using background clutter modeled on that used in perceptual psychology experiments [Cutting *et al.*, 1988]; using gait samples with variation in tempo; using gait samples from movement which is not exactly parallel to the image plane. But the experiments reported here demonstrate, first, that the architecture is capable of discrimination of human gait within one cycle of the gait, and second, that it exhibits tolerance to clutter while retaining the ability to reject stimuli superficially similar to modeled gaits.

# 7 Conclusions and Future Work

## 7.1 Summary

Our goal in this dissertation was to provide a computational account of human interpretation of Johansson's Moving Light Displays, consistent with the scientific facts and working within the structured connectionist paradigm. Considering that no such account has been given to date by the computer vision community, our intention was to demonstrate the power of structured connectionism as an explanatory tool. We took as our point of departure the framework provided in [Feldman, 1985] and [Feldman, 1988] and partially elaborated in [Olson, 1989]. We analyzed the tasks involved and delimited a problem amenable to solution: discrimination of 400 millisecond presentations of Moving Light Displays generated by a single articulated object moving parallel to the image plane. We assumed solutions to some of the low-level tasks: correspondence, segmentation and tracking. We argued that some high-level tasks, such as viewpoint recover, verification and planning eye-movements, could not be significantly involved in the solution to our problem. The core computational issues remaining to be addressed were: structure-from-motion, feature composition, representing and indexing high-level models of movement, the use of attention, and gestalt formation.

The dissertation presented an architectural model for some of the intermediate and high level processes which might be involved in the recognition of Moving Light Displays. We have implemented the architecture (absent the structure-from-motion component) and tested it on three samples of each of three gaits executed by each of four individuals. We have shown that the architecture exhibits tolerance of variation in spatial scale, the ability to reject stimuli superficially similar to modeled gaits, translation invariance (both spatial and temporal), and tolerance of static background clutter. We have also shown sensitivity to dynamic background clutter consisting of a gait different from the target, and sensitivity to rotation of the display in the image, both of which are consistent with perceptual data [Sumi, 1984; Cutting *et al.*, 1988].

The architecture employs multiresolution feature hierarchies which index hierarchical central representations of gait called scenarios. It incorporates a parallel attentional mechanism which combines low-level feature data and high-level contextual information to guide

the focusing of attention on particular spatial locations, features, and scenarios. We have shown the importance of the attention mechanism even for dealing with a single-object scene, the importance of competition within the attention mechanism, and the advantages of organizing the scenarios hierarchically.

## 7.2 Contributions

We have tried to take a multi-disciplinary approach, focusing on computational considerations but attempting to produce a cognitive model of potential use to brain scientists. We discuss the contributions this work makes in three fields of research: Computer Vision, Connectionist Modeling, and Cognitive Science.

### 7.2.1 Contributions in Computer Vision

This dissertation makes several contributions in the field of computer vision. First, we demonstrate a solution to a core problem - recognition of articulated motion sequences - that no existing techniques have addressed adequately. Second, we provide a task analysis for the problem and a set of representations and mechanisms for the solution that are implementation independent. Third, in the classical formulation of recognition problems as "what" and "where", we extend the conceptualization of "what" to include "what's happening"; and show that it is not necessary to solve "what" and "where" before solving "what's happening". Fourth, we demonstrate that solving the "what" and "where" problems is possible without the recovery of precise shape and motion information. Fifth, we introduce a novel attentional mechanism based on temporal simultaneity that is shown to enable cooperation between the "what's happening" and "where" processes. This also demonstrates the use of high-level contextual information, as well as low-level data, to guide the spatial and conceptual foci of attention.

#### Recognizing Human Motion

Recognition of motion sequences is a crucial ability for biological and robot vision systems. In Chapter 2 we reviewed the related work in Computer Vision. Although there has been prior work on recognition of complex non-fluid motion (e.g., [Badler, 1975; O'Rourke and Badler, 1980; Tsotsos *et al.*, 1980; Tsotsos, 1987; Gould and Shah, 1989; Allmen and Dyer, 1990]), none of it has come close to providing a solution, or even a framework for a solution, to the problem of recognizing human motion. Our implemented architecture demonstrates a solution in the domain of moving stick figures, assuming the work in [Olson, 1989] to derive the input we need, and the entire structure is within a consistent framework [Feldman, 1985; Feldman, 1988]. The structure-from-motion task was not addressed, but there are existing solutions in the computer vision literature for our domain [Rashid, 1980; Hoffman and

Flinchbaugh, 1982], although it would take some effort to recast these within Feldman's framework. We discuss the structure from motion problem further in section 7.3.4.

### General Task Analysis and Mechanisms

In the introduction we discussed the various processes involved in recognition of Moving Light Displays, and delimited a research problem which, we argued, is primarily dependent on a limited subset of those processes. Our task analysis was independent of the eventual implementation in a connectionist network. Moreover the architecture, and even many of the details of the representations and processing mechanisms, can be recast in conventional vision terms. The feature hierarchies can be thought of as a hierarchy of Hough transforms [Ballard, 1984], with spatial composition performed by a soft AND function. In the Hough transform formulation, the O-binding mechanism becomes a pre-pass to decide how many votes to allocate to each recipient bin, rather than equal allocation which is the standard Hough technique. The scenarios are a type of state-interval-state representation [Marr and Vaina, 1982], using confidence factors to replace unit activations. An entire scenario can be considered a knowledge-frame [Minsky, 1980] with slots for the states, clocks (intervals) and for overall confidence. The what/where attention map is a representation of what is happening, which is used to increase the vote count in particular top-level features and to increase the weight of votes cast by particular features for particular scenario states. Its competitive nature embodies the idea that only one action can occupy one portion of the image at any one time.

### "What" and "Where" not Prior to "What's Happening"

Computer vision work has distinguished the "What" problem (i.e., what is in the scene) from the "Where" problem (i.e., where is a particular entity in the scene). We introduce a third visual problem, "What's Happening", which addresses the question of identifying changes in the scene. This problem has received attention at a low level, in recovering motion parameters of rigid and non-rigid objects. To the extent that it has been addressed at a high level, most previous work has assumed that solutions to the "What" and "Where" problems are a pre-requisite to solving the "What's Happening" problem. Our work shows that, for scenes containing an articulated non-rigid object in complex motion (e.g., a person walking), the solution to the "What's Happening" problem need not wait for solutions to the "What" and "Where" problems.

### Metrical Shape and Motion not Required

Recovery of precise shape and motion information is not required for generating solutions to the "What's Happening" problem. Generally this means that, for scenes with complex structured changes, precise recovery of shape and motion information is not required to

generate solutions to the "What" or "Where" problems (although precise metrical information may well be needed to verify those solutions). We can make progress in high-level vision without making unrealistic assumptions about what lower-level processes can extract from an image sequence.

**Data/Context Driven Parallel Attention**

The what/where attention map is a significant advance in conceptualization of attention in the area of computer vision. This is demonstrated by four distinguishing features:

- **Context and Data Driven.** The mechanism uses both low-level image data and high-level interpretations in order to focus attention.
- **Concepts and Locations are Attended.** The mechanism can focus attention on particular conceptual entities (scenarios in our case) as well as particular spatial locations.
- **Parallel Operation.** The mechanism attends to multiple locations and entities in parallel, although entities compete for locations, thus disallowing more than one entity to focus on a particular location at a given time.
- **Simultaneity as Cue.** The mechanism uses resonance over time of internal representations with external events to generate cues for attention.

Although psychologists have long had a broad view of attention [Shiffrin, 1988], there has been little interest until recently in the computer vision community. The conceptualization has been one of sequential search, either overt [Rimey and Brown, 1990] or covert [Ahmad and Omohundro, 1991], with processing restricted to focusing on a particular spatial location or region. We have provided a new view of attention as a process or set of processes which can focus on both locations and conceptual entities and do so in parallel. The mechanism uses temporal simultaneity of internal expectations and external events as a cue for attention, and is therefore restricted to dynamic scenes.

### 7.2.2 Contributions in Connectionist Modeling

The major contribution of this dissertation in the field of Computer Science is its elaboration and demonstration of mechanisms useful in Massively Parallel Processing. There are four mechanisms of primary significance and originality. First, we introduced a new representation for temporal sequences, the scenario, which is quite distinct from other connectionist work on sequence [Pearlmutter, 1990]. Second, we have shown how a topographic attention map can be used to form perceptual gestalts between spatially-indexed and central representations, and how it can be used to focus bottom-up activation and top-down priming flow. Third, we proposed and demonstrated the use of temporal resonance

between internal activation flows and external events as a cue for attention and binding. Fourth, we have shown how the temporal nature of slow parallel hardware can be used to advantage in perceptual processing, via the representation of external time by internal processing and communication delays.

### Scenario Representation of Temporal Sequences

The representation of temporal sequences embodied in a scenario is a step forward in research on representation of sequence and time explicitly in network structures. Although the work on Time-Delay Neural Networks [Waibel, 1989] does use explicit delays in links to represent time, the aim there is to concatenate appropriately delayed signals in a buffer, and then use (static) pattern recognition techniques on the buffer. Our interval unit represents a flexible and, of great importance, scalable time interval. The magnitude of the interval can be scaled, although we did not use this feature in our implementation. The time interval represented by the unit can be a soft or hard constraint on scenario processing, according to whether the slope of the ramps up and down are shallow or steep. We showed how the slope can be dynamically varied to increase the severity of temporal constraints as confidence levels rise. The scenarios can be composed in part/whole hierarchies, which gives rise to more coherent resonance with the input data. The scenario representation is suitable for any application in which the temporal sequence is naturally divided by discrete events into discrete states. This does not rule out all continuous signals - human gait is a continuous motor activity - but it does make the representation most appropriate for hierarchically structured motion.

### Attention and Binding in Topographic Maps

We have extended Olson's feature-binding mechanism [Olson, 1989] to enable gestalt formation between spatially-localized and central representations. The proxies are used as representatives at particular locations of particular scenarios in the competition to bind features in a gestalt. They are also the basis for the attention map. This is the first connectionist attention mechanism that integrates bottom-up and top-down cues in its control strategy, and also the first to apply attention in parallel to diverse spatial locations and high-level concepts.

### Temporal Correlation and Time as its own Representation

The use of temporal simultaneity as a cue for binding recalls recent work in neurophysiology (for example, [Crick, 1984; von der Malsburg, 1987; Gray and Singer, 1989]) and neural networks (for example, [Hummel and Biederman, 1990; Shastri and Ajjanagadde, 1990]), but these works use synchronicity in neural spike trains as the basis for binding. Our mechanism works at much longer time scales related to the rate of change in the perceptual environment. The idea of establishing synchronicity between an internal representation and

external events in the world is fundamentally different to that of establishing synchronicity between two internal representations (in the former case, only one representation is under the control of the perceptual system, in the latter case both are). Thus time itself is elegantly employed to implement constraints on recognition.

### 7.2.3 Contributions in Cognitive Science

Many of the advances noted above are of relevance to Cognitive Science. Here we elaborate the contributions that this dissertation makes from the Cognitive Science vantage point.

#### Open Problem Modeled

Since Johansson first demonstrated the remarkable ability of humans to discriminate gait from Moving Light Displays[1] [Johansson, 1973], cognitive scientists have attempted to model the phenomenon (usually without a computational implementation). Johansson himself proposed "visual vector analysis" [Johansson, 1973]. Other attempts have been made to model some of the processes involved (e.g., [Hoenkamp, 1978; Cutting *et al.*, 1978; Cutting, 1981b]), but there has been no satisfactory explanation of how gait could be represented and how gait discrimination is achieved.

#### Cognitive Task Analysis

This dissertation provides a detailed analysis of the tasks involved in MLD discrimination. Any cognitive model would have to follow an analysis similar to this. The problems of representing temporal sequences, of developing a hierarchy of composed features, of using spatially distributed features to index central gait representations, and of using an attentional process to focus processing, are all ones that would have to be solved in a cognitive model. In addition, given the slow switching time of neurons, a cognitive model must be massively parallel or obviously parallelizable.

#### Synergistic What and Where Processes

The distinction between "what" and "where" processes in cognition, and corresponding pathways in brain architecture is directly modeled in our architecture. The scenarios are a representation of "what" (or rather, "what's happening"), while the attention map is a representation of "where". We have demonstrated cooperative "what" and "where" processes, showing not only how each process functions but also that they can interact with synergy.

[1] Usually known as "biological motion" or "point light walker" in the psychology literature.

## New Brain-Modeling Ideas

The new mechanisms described above in the section on contributions in massively parallel processing may be suggestive to brain-modeling scientists of how certain functions might be performed in the brain. The scenarios provide a representation of temporal sequence using discrete states and parameterized time intervals, which could be used in modalities other than vision, principally audition and motor control. The attention map, with its cues based on temporal simultaneity, is a new example of an attention mechanism that is consistent with neural processing. It is possible that some of the mechanisms could help to explain the neurophysiological data on recognition of Moving Light Displays [Chitty *et al.*, 1987]. Overall, the architecture presented in this dissertation is the most detailed elaboration to date of a model of high-level motion processing consistent with the scientific facts.

## Perceptual Experiments

Perhaps the most useful aspect of this work for cognitive scientists is that we have developed a model that makes predictions, suggests experiments, and is open to further refinement as experimental data dictate. For cognitive modeling work beyond the most general levels, a detailed, implemented computational model is imperative. Interactions of components and mechanisms of the model cannot always be foreseen, so that implementation and testing is required. Articulation of details of the model helps to identify the core issues and trade-offs that must be made, and to show up assumptions that can provide experimental predictions. Several experiments are suggested, for several hundred msec presentation of two-dimensional MLDs:

1. Reverse time:

   - Is running backwards identifiable?
   - Is there a difference between reversed movements that are real-world possible (e.g., walking backwards) and those that are not (e.g., skipping backwards)?

   The model predicts that unfamiliar or impossible movements would not be identifiable, since there would be no internal model of the movement. It would be interesting to see if identification is possible if the presentation time is increased.

2. Spatial rotation (e.g., upside-down) [Sumi, 1984]:

   - is the gait identifiable?
   - is the object identifiable? Compare this with rotated stick figures.

   The model predicts that gait would not be identifiable without enough practice to learn discriminating features. The results for object identification would indicate the ability to recover structure from motion in a manner useful to the discrimination process.

3. Multiple objects/gaits/phases in the scene simultaneously, overlapping or separated:

   - several different objects and different gaits.
   - identical objects but different gaits.
   - identical objects and gaits, but different phases.

   The model makes the strong prediction that with spatially separated presentation of different objects and/or gaits, all would be identifiable. Human capability is unlikely to be this unlimited, and the results would help to pin down the limitations on the attention mechanism. It predicts that with short presentation of two objects at separated locations executing the same gait but out of phase with each other, only one of the phases would be identifiable (since there is only one scenario for each gait), or perhaps two sufficiently widely separated phases. Longer presentation would allow sequential attention to isolate each object and movement. The model predicts that overlapping objects would produce interference (masking), as was found in [Cutting *et al.*, 1988].

4. How important is translational motion of the entire figure [Proffitt, 1988]:

   - horizontally in direction of motion.
   - vertically (e.g., up and down during running).

   The model predicts no effect of translational motion. Informal presentation in our laboratory suggests that for some movements, vertical motion is a strong cue for object and/or gait. The model's prediction is unlikely to be fully borne out, but at least for 400 msec presentation times we would expect the effect of translational motion to be small.

5. Manipulate the MLD so that the inter-frame time/space interval is out of bounds for the apparent motion effect. Repeat with MSDs (moving stick displays). The results would indicate how dependent the recognition process is on motion information. The model predicts that performance should drop dramatically, although perhaps not to zero since the scenarios are indexed to some extent by shape features. The results would also show to what extent shape information is recoverable from the MLD when only static information is available to the structure-from-motion process.

6. Occlusion: How important is it to have the distal leg/arm disappear from view when occluded by the body [Proffitt, 1988]? Is a wire-frame figure just as recognizable? The model assumes a wire-frame figure, and it would be interesting to find out what effect addition of occlusion cues has on the discrimination task. Informal presentation in our laboratory indicates that occlusion cues contribute to positive judgments of realism, but it is not clear that discrimination of 400 msec stimuli is aided.

7. Moving dots away from their correct position (e.g., moving the elbow dot up the arm towards the shoulder), or removal of dots. Anecdotally, both of these are said to severely degrade performance. The model makes no prediction, but this experiment would provide additional data on the recovery of structure from motion (see section 7.3.4).

8. Eye tracking during first 1000 milliseconds. Is there any systematic location or sequence of locations that are attended? If the image is stabilized, is discrimination affected? The model predicts that stabilization should have no effect on the discrimination of 400 msec MLDs.

### 7.2.4 Generality and Extensibility

Several software tools were constructed to ease the process of conversion of WATSMART gait data to forms suitable for the networks to use. These tools would make it relatively easy to process new WATSMART data samples were they to be collected; whether new samples of the three gaits or samples of different gaits. A program combined the samples into specifications of scenarios, one for each gait, and wrote them to file (see Appendix A). The network construction software read a data file specifying in propositional form the structure of the biped and built an internal object-oriented representation of the stick-figure. Then the network construction software read the scenario specification files, which referred to parts of the stick figure. It would be easy to extend the implementation to cover other stick-figures (e.g., animals) by simply construction specification files for the structure of the animals. The labor-intensive work that needs to be done, after gait samples have been collected, is determination of scenario events. This is aided by overlaying screendumps of the gait samples (e.g., Figure 6.1), but selection of events is not automated.

For the implementation we used a modified version of the Rochester Connectionist Simulator [Goddard *et al.*, 1988]. The simulator was adapted so that network construction code and unit functions could be written in C++ and the unit functions dynamically updated as necessary. The use of an object-oriented language for network construction was a major advantage, given the highly structured nature of the network. The ability to have different unit classes with their own methods for handling state variables, update functions and graphical display made debugging much easier than it would otherwise have been.

The major lesson of the implementation and debugging process was that support for structured graphic display is essential for structured connectionist networks. In an architecture as complex as the one described here - which is, after all, about as simple as a cognitive model could be - the ability to structure the graphic display to reflect the structure of the network is crucial. We were able to build our own graphical display structures, using the symbolic models (object-oriented) of a biped and the architectural components. This allowed us to hierarchically traverse the biped model and display units related to single limb-segments, limbs, pairs of limbs, or the entire biped. Pop-up topographic maps allowed us to specify which locations in the feature maps were to be displayed. Different

unit classes contained their own methods to graphically display different aspects of themselves. The highly structured and customized display was made possible by the use of an object-oriented language (C++), a compiler with associated symbolic debugger (G++ and GDB), a general window system (X11) and the highly flexible and customizable Rochester Connectionist Simulator. However, the simulator graphics provided only minimal support for a structured display (multiple display windows, multiple unit icons and aspects, multiple display modes). We believe that there is a need for a simulator designed around an object-oriented language, with support for the specification and examination of highly structured networks, possibly characterized as a symbolic system, and with graphic display capabilities integrated into the structuring of the networks.

The main lesson learned from the experiments in terms of simulator capabilities was that even a simple cognitive model, such as the one presented in this dissertation, severely strains the computing resources available on today's workstations. In terms of memory use, the simple experiments with one gait sample could be run on a machine with 16 Mb of memory, while the dual sample trials required 24 or 32 Mb machines. There would have been no chance of modeling the structure-from-motion process. In terms of CPU use, we were able to run a single experiment over several nights using a network of about 20 16 MIP machines. Parameter changes and bug fixes entailed re-running experiments. Our conclusion is that in order to investigate connectionist cognitive models in realistic domains, we need supercomputer-level resources. The ideal setup would be to have simulation environments that run on both workstations and supercomputers, presenting the same user-level interface on all machines. Developing and debugging of models could then be done on workstations, with an easy upward path to full-scale testing of the models.

## 7.3 Future Work

The major goal of modeling human interpretation of Moving Light Displays has been achieved. Along the way, we have developed some new and interesting mechanisms that may be of use in computer vision, connectionist modeling and cognitive science. However, there are a number of deficiencies in the architecture and implementation which suggest interesting avenues of research.

### 7.3.1 Variable Tempo

The scenarios as described are parameterized for different tempos - the interval units can have their time constants scaled by an external input. But we did not propose any mechanism for determining how that tempo input to the interval units could be determined. There are two components to this problem:

- Tracking: assuming the scenario is resonating with the input gait, but the tempo of the input is changing slowly, how can we arrange to have the tempo parameter track

the input tempo?

- Initialization: at the commencement of presentation of the input, how can we quickly determine a value for the tempo parameter that is likely to be close to the true value?

If solutions to both these problems can be found, then we will have an architecture that is able to recognize many different examples at different tempos as the same gait, and which is also be able to indicate the tempo. We have rough ideas for how to solve both these problems.

### Tracking

Associate with each scenario a tempo unit, which computes the current tempo. It receives as input pairs of links, each pair arriving at a different site. Each pair has one link from an event unit, and the other from the preceding interval unit. If the tempo parameter matches the input tempo then the event unit will provide a spike of activation (event enabled) during the peak plateau phase of the interval unit. If the tempo parameter is close to but not quite identical with the input tempo, then the event unit spike may occur during the ramp-up or -down of the interval unit. If this happens, the tempo unit adjusts its output slightly upwards (tempo increased) if the spike occurred on the ramp-up, or slightly downwards (tempo decreased) if the spike occurred on the ramp-down. This mechanism should track slowly changing tempo, but it will not work if the tempo changes too fast (spike occurs before ramp-up or after ramp-down), or if the peak plateau phases in some interval units are long and others short (the tempo may have changed a lot before the spike hits a ramp). Both of these problems are mitigated if the ramps are shallow, which is the case when the scenario is not very active. Therefore, we can hope that the mechanism will work for indexing and tracking slowly changing tempo.

### Initialization

Use the output of low-level velocity units to determine an initial value for the tempo. The idea here is that if, for example, tempo is doubled, then all angular (and translational) velocities are doubled. If we take the average magnitude of these velocities and compare it to a canonical value for canonical tempo, then we shall have a good initial estimate of tempo. There is a complication in that, for human gait, the velocities reach a minimum at maximum extension of the limbs (i.e., legs far apart), and a maximum at minimum extension. To deal with this problem we could use accelerations instead of speeds, since greater accelerations would occur for faster tempo at all phases of the gait. Alternatively, we could use an estimate that depends on a combination of extension and speeds. The problem with using accelerations is that second derivatives of noisy signals are not very reliable. The problem with combining extension into the estimate is that it presupposes an interpretation of the scene.

### 7.3.2 Qualitative 3D

Early in this work we decided to focus on two-dimensional processing. In our informal displays of MLDs to co-workers, we found that occlusion cues (e.g., the distal knee dot disappears when behind the proximal knee) were important for developing the feeling of realism. Experimental data [Proffitt, 1988] indicated that occlusion cues can make a big difference in the ability to correctly identify an MLD as a walking biped, although in these experiments the MLD was computer generated. In order to handle this type of cue, we would need assembly-level features incorporating the occlusion. The scenarios would have additional events enabled by onset and offset of occlusion. Some mechanism would need to be developed to handle the fact that discrimination is possible with and without occlusion cues; perhaps interval units that skip an event or two. This use of qualitative 3D (ordinal depth information) is quite different from the problems of full 3D motion. It is essentially the same underconstrained indexing problem that is widespread in dealing with 2D images.

### 7.3.3 Learning

Although much of the work in neural networks has been focused on learning, it is not an issue at the forefront of research in the structured connectionist paradigm. However, we found in this thesis work some of the limits of hand coding networks, particularly mapping networks. We believe that some of the gradient descent learning algorithms (backpropagation or one of its variants) could be used to learn the mappings between levels in the feature hierarchies, if we know what features we want at the assembly level. The question of what assembly-level features we need is not easy to answer. To a large extent it depends on the specification of event enablement and support in the scenarios. In this work, we visually examined the gait data, appropriately displayed, and hand-selected the events. While there has been a significant amount of neural network research on learning sequences, our scenario representation is beyond the reach of current learning technology. It is possible that the event states could be learned using, for example, [Cleermans *et al.*, 1989], and there is work on learning time-delays [Bodenhausen and Waibel, 1991] which might be of use for specifying the interval unit parameters. Putting these ideas and others together into a framework sufficient to learn scenarios is a matter for future research.

### 7.3.4 Structure from Motion

When we began this study, we had hoped to elaborate that part of the architecture which transforms dot motion into activations of line segment units. This is the structure-from-motion problem for MLDs. It has been studied quite extensively in the computer vision community (e.g., [Rashid, 1980; Webb and Aggarwal, 1982; Hoffman and Flinchbaugh, 1982]). However these algorithms do not model the data in the psychology literature. They neither mirror the abilities of the human visual system (e.g., [Braunstein, 1986; Doner *et al.*, 1984; Todd, 1985]), nor do they exhibit the non-veridical percepts generated

by the human visual system (e.g., [Braunstein, 1986; Dosher *et al.*, 1986; Todd *et al.*, 1988]), although Rashid's system could be part of a true cognitive model. We developed a modified Hough Transform network using Olson's trajectories [Olson, 1989] as input and producing static and moving line segment information as output. It was designed as a single process to work with MLDs or MSDs (moving stick displays), therefore being more general than the work cited above. However, testing of the network revealed that a voting network could not solve the problem, even with the trajectories as input[2]. Two major problems where found.

### Trajectory Continuity

During human gait, there are periods of significant length during which a limb segment does not rotate and/or translate. For example, during walking, the swing-leg swings forward, and eventually the thigh stops moving (more or less) while the calf keeps rotating about the knee. When the calf finally stops moving, the foot is planted on the ground and the leg enters its support phase. During the time that the thigh is more or less stationary, the knee dot is moving so little as to produce no reliable trajectory, and therefore the voting network no longer votes for the line segment representing the thigh. There are significant periods of time during which one or several limb segments is thereby "lost". We need some mechanism that allows a previous interpretation (e.g., this dot and that dot are connected) to persist for a hundred milliseconds or so even in the absence of supporting evidence. A simple solution would be to endow the segment level units with hysteresis, so that once activated they decayed slowly. A more sophisticated version would be to have higher level features (e.g., leg-like components) provide feedback activation to the segment level units, but this would have to be done carefully to avoid uncontrolled hallucination.

### Reference frames

We had assumed that limb segments could be found by the voting network because each dot rotates about another dot, and the MLD is assumed to be tracked. The tracking assumption is important: it transforms the hip and shoulder dot movement to be close to zero, and the knee and elbow dot motion to be close to pure circles about them. The voting network used the trajectory coding to find dots near the expected center of curvature. The wrist and ankle dots are executing epicycles, but we had thought that during much of the motion the coarseness of the trajectory coding would result in the correct segment level units being activated, due to the fact that the direction of the center of curvature would be approximately the same as the line connecting the dots (for example, for much of the arm-swing while walking, the wrist is executing an approximate circle about the shoulder, while the elbow is close to the line joining wrist and shoulder). It turns out that in many crucial phases of

[2]Since the trajectories code for location, approximate radius of curvature and approximate direction of the center of curvature, they already contain in explicit form the information needed to connect a dot moving in a circle to a dot at the center of the circle.

motion, the wrist and ankle dot trajectories are not even close to what is needed to vote for the true connecting segments (forearm and calf). However, there are two ways in which the veridical connecting segments could be found, both of which are related to frames of reference. First, in the frame of reference attached to the knee, the ankle is moving in a perfect circle (this is Johansson's visual vector analysis). Second, if we apply the rule that dots with substantial common translatory motion are probably in the same frame of reference and therefore probably connected, then we can hypothesize the knee/ankle connection for some parts of the gait during which there is no rotation of the ankle about the knee but, nonetheless, significant motion of the ankle with the knee (e.g., during that portion of the swing phase when the calf reaches its greatest angular distance from the vertical). This rule also would allow us to relax the assumption that the figure is being tracked: the knee and elbow dots would be seen to be moving in the frame defined by the hip and shoulder respectively.

The solution to both these problems, we believe, is predicated on top-down feedback. Informal experiments conducted in our laboratory indicate that 400 msec presentation of MLDs representing arbitrary articulated figures, of similar complexity to a biped, often does not give rise to a veridical percept. We believe that the ability to perceptually organize the moving dots into a structure of connected moving line segments is dependent on high level models of articulated objects and how they can move. A general model of the recovery of structure from motion in the human visual system would be an overly ambitious goal at this stage, but a cognitively plausible solution to the structure-from-motion problem for 400 msec MLDs is approachable and would bring us a step closer to a solution for the general problem.

Our current notion of how recovery of structure from motion could be done is as follows. The scheme consists of four interacting processes together with the higher level feature and scenario networks discussed in this dissertation. The **rotation** process is a Hough network that uses trajectory input to hypothesize line segments, by recovering line segments due to motion in a circle about a stationary point. The **translation** process is based on velocity and recovers line segments due to two dots having similar velocities. These are the bottom-up processes. The line segments begin to activate higher level features of all kinds, and to index models of objects and how they move. These features and models provide feedback to the segment level (perhaps using O-binding). The **continuity** process uses the feedback to maintain existing segments if they have top-down feedback support and recently had bottom-up image support. The **context** process is intended to implement Johansson's visual vector analysis. It uses the top-down feedback to influence the rotation process. An active component-level feature would activate local frames of reference for its constituent segment-level features. This would be done by modifying the voting pattern in the rotation process for a small spatial area in which the segment should occur. For example, a leg-like feature would activate a local frame of reference attached to the knee. In this local frame of reference the ankle dot is traveling in a circle, so that the rotation process will recover the calf segment. The activation of component-level features would be possible because the limb-segments attached to the body (thigh and upper-arm) will be recovered bottom-up

by the rotation process, because either the MLD is tracked, or the context process uses the overall translational motion of the cloud of dots to set the first reference frame. In either of these cases, the elbow and knee dots will be moving in circles and so the rotation process will recover the upper-arm and thigh segments without feedback. We hope that this might be sufficient to start activating component-level features and possibly even scenarios, thereby creating the semantic feedback which the context process needs to function.

These ideas are not fleshed out, but they are sufficiently developed that the four structure-from-motion processes and associated representations could be designed, and the work in this thesis provides the basis for the essential knowledge-dependent feedback. We believe this to be a very exciting line of research, and one that would generate many predictions that could be psychophysically tested, along the lines undertaken in [Cutting, 1981a].

## 7.4 Conclusion

In this dissertation, our aim has been to understand, from a computational perspective, human interpretation of articulated motion. We have presented an architectural model of the intermediate and high-level processes, and have demonstrated via the implementation that the ideas are computationally sound. Structured Connectionism has proven to be an effective means for modeling recognition of articulated motion. Some new connectionist modeling techniques have been developed and explored in the process. Although the model has many rough edges, there is no other model of high level motion processing consistent with the scientific data. It has been difficult to balance the competing claims of computational elegance and clarity, psychological relevance, and neurophysiological plausibility, but the effort has been worthwhile. We hope that this dissertation provides not only a framework for further research - particularly in structure-from-motion and other perceptual problems of a time-varying nature - but also an example of how computational modeling and cognitive science can interact fruitfully.

# Bibliography

[Ahmad and Omohundro, 1991] Subutai Ahmad and Stephen Omohundro, "Efficient Visual Search: A Connectionist Solution," In *Proceedings of the 13th Annual Conference of the Cognitive Science Society*, Chicago, 1991.

[Allmen and Dyer, 1990] Mark Allmen and Charles R. Dyer, "Cyclic Motion Detection Using Spatiotemporal Surfaces and Curves," In *Proceedings of the International Conference on Pattern Recognition*, pages 365–370, 1990.

[Almeida, 1987] L. B. Almeida, "A Learning Rule for Asynchronous Perceptrons with Feedback in a Combinatorial Environment," In M. Caudill and C. Butler, editors, *IEEE First International Conference on Neural Networks*, volume 2, pages 609–618, San Diego, 1987.

[Aloimonos and Shulman, 1989] John (Yiannis) Aloimonos and David Shulman, *Integration of Visual Modules*, Academic Press, San Diego, CA, 1989.

[Badler, 1975] Norman I. Badler, *Temporal Scene Analysis: Conceptual Descriptions of Object Movements*, PhD thesis, Department of Computer Science, University of Toronto, 1975.

[Bailey and Hammerstrom, 1986] Jim Bailey and Dan Hammerstrom, "How to make a billion connections," Technical Report OGC CS/E-86-007, Oregon Graduate Center, Beaverton, OR, 1986.

[Ballard, 1984] Dana H. Ballard, "Parameter Networks: Towards a Theory of Low-Level Vision," *Artificial Intelligence*, 22, 1984.

[Ballard, 1986] Dana H. Ballard, "Cortical connections and parallel processing: structure and function," *The Behavioral and Brain Sciences*, 9(1):67–120, March 1986.

[Ballard, 1990] Dana H. Ballard, "Animate Vision," Technical Report 329, Department of Computer Science, University of Rochester, February 1990.

[Ballard and Ozcandarli, 1988] Dana H. Ballard and Altan Ozcandarli, "Eye Fixation and Early Vision: Kinetic Depth," In *Second International Conference on Computer Vision*, pages 524–531, December 1988.

[Bassili, 1978] John N. Bassili, "Facial Motion in the Perception of Faces and of Emotional Expression," *Journal of Experimental Psychology: Human Perception and Performance*, 4(3):373–379, 1978.

[Bodenhausen and Waibel, 1991] Ulirich Bodenhausen and Alex Waibel, "The Tempo 2 Algorithm: Adjusting Time-Delays by Supervised Learning," In Richard P. Lippmann, John E. Moody, and David S. Touretzky, editors, *Advances in Neural Information Processing 3*, pages 155–161. Morgan Kaufmann, San Mateo, CA, 1991.

[Bourlard and Wellekens, 1988] H. Bourlard and C. J. Wellekens, "Links between Markov Models and Multilayer Perceptrons," Technical Report M 263, Philips Research Laboratory, Brussels, September 1988.

[Braunstein, 1976] M. L. Braunstein, *Depth perception through motion*, Academic Press, New York, 1976.

[Braunstein, 1986] M. L. Braunstein, "Perception of rotation in depth: the psychophysical evidence," In N. I. Badler and J. K. Tsotsos, editors, *Motion: Representation and Perception*, pages 242–247. North-Holland, New York, 1986.

[Califano *et al.*, 1989] A. Califano, R. M. Bolle, and R. W. Taylor, "Generalized Neighborhoods: A New Approach to Complex Parameter Feature Extraction," In *Proceedings of the Computer Society Conference Conference on Computer Vision and Pattern Recognition*, June 1989.

[Chitty *et al.*, 1987] Andrew J. Chitty, David I. Perret, Amanda J. Mistlin, and M. Harries, "Visual cells sensitive to biological motion," Technical report, Department of Psychology, University of St. Andrews, 1987.

[Chou, 1988] Paul B. Chou, "The Theory and Practice of Bayesian Image Labeling," Technical Report TR 258, Department of Computer Science, University of Rochester, August 1988.

[Chun, 1986] Hon Wai Chun, "A representation for temporal sequence and duration in massively parallel networks: Exploiting Link Interactions," In *Proceedings of AAAI-86*, August 1986.

[Cleermans *et al.*, 1989] Axel Cleermans, David Servan-Schreiber, and James L. McClelland, "Finite-State Automata and Simple Recurrent Networks," *Neural Computation*, 1:372–381, 1989.

[Cooper, 1989] Paul R. Cooper, *Parallel Object Recognition from Structure (The Tinkertoy Project)*, PhD thesis, University of Rochester, 1989.

[Cottrell, 1985] Garrison Weeks Cottrell, *A connectionist Approach to Word Sense Disambiguation*, PhD thesis, Department of Computer Science, University of Rochester, 1985.

[Crick, 1984] F. H. C. Crick, "The function of the thalamic reticular spotlight: The searchlight hypothesis," *Proceedings National Academy of Sciences USA*, 81:4586–4590, 1984.

[Cutting, 1981a] James E. Cutting, "Coding Theory Adapted to Gait Perception," *Journal of Experimental Psychology: Human Perception and Performance*, 7(1):71–87, 1981.

[Cutting, 1981b] James E. Cutting, "Six tenets for event perception," *Cognition*, 10:71–78, 1981.

[Cutting *et al.*, 1988] James E. Cutting, Cassandra Moore, and Roger Morrison, "Masking the motions of human gait," *Perception and Psychophysics*, 44:339–347, 1988.

[Cutting and Proffitt, 1982] James E. Cutting and Dennis R. Proffitt, "The Minimum Principle and the Perception of Absolute, Common and Relative Motions," *Cognitive Psychology*, 14:211–246, 1982.

[Cutting *et al.*, 1978] James E. Cutting, Dennis R. Proffitt, and Lynn T. Kozlowski, "A Biomechanical Invariant for Gait Perception," *Journal of Experimental Psychology: Human Perception and Performance*, 4(3):357–372, 1978.

[Cutting and Kozlowski, 1977] J.E. Cutting and L.T. Kozlowski, "Recognizing friends by their walk: Gait perception without familiarity cues," *Bull. Psychonometric Soc.*, 9(5):353–356, 1977.

[Derthick, 1988] Mark Derthick, *Mundane Reasoning*, PhD thesis, Carnegie Mellon University, Pittsburgh, PA, 1988.

[Doner *et al.*, 1984] J. Doner, J. S. Lappin, and G. Perfetto, "Detection of three-dimensional structure in moving optical patterns," *Journal of Experimental Psychology: Human Perception and Performance*, 10:1–11, 1984.

[Dosher *et al.*, 1986] B. A. Dosher, G. Sperling, and S.A. Wurst, "Tradeoffs between stereopsis and proximity luminance covariance as determinants of perceived 3D structure," *Vision Research*, 26:973–990, 1986.

[Dyer, 1987] Charles R. Dyer, "Multiscale Image Understanding," In Leonard Uhr, editor, *Parallel Computer Vision*. Academic Press, Orlando, Florida, 1987.

[Elman, 1988] Jeffrey L. Elman, "Finding Structure in Time," Technical Report 8801, Center for Research in Language, University of California, San Diego, 1988.

[Fahlman, 1991] Scott E. Fahlman, "The Recurrent Cascade Correlation Architecture," In Richard P. Lippmann, John E. Moody, and David S. Touretzky, editors, *Advances in Neural Information Processing 3*, pages 190–196. Morgan Kaufmann, San Mateo, CA, 1991.

[Feldman, 1985] Jerome A. Feldman, "Four frames suffice: A provisional model of vision and space," *The Behavioral and Brain Sciences*, 8:265–289, 1985.

[Feldman, 1988] Jerome A. Feldman, "Time, Space and Form in Vision," Technical report, Department of Computer Science, University of Rochester, 1988.

[Feldman and Ballard, 1982] Jerome A. Feldman and Dana H. Ballard, "Connectionist Models and their Properties," *Cognitive Science*, 6:205–254, 1982.

[Feldman *et al.*, 1988] Jerome A. Feldman, Mark A. Fanty, Nigel H. Goddard, and Kenton Lynne, "Computing with Structured Connectionist Networks," *Communcations of the ACM*, February 1988.

[Fox and McDaniel, 1982] Robert Fox and Cynthia McDaniel, "The Perception of Biological Mottion by Human Infants," *Science*, 218:486–487, October 1982.

[Frazier, 1990] Gary Frazier, "*Ariel* : A 100 GigaFlops Simulator for Connectionism Research," Unpublished paper, 1990.

[Freyd, 1983] Jennifer J. Freyd, "The mental representation of movement when static stimuli are viewed," *Perception and Psychophysics*, 33(6):575–581, 1983.

[Geman and Geman, 1984] Stuart Geman and Donald Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, 1984.

[Gibson, 1950] James J. Gibson, *The Perception of the Visual World*, Boston: Houghton Mifflin Company, 1950.

[Goddard, 1989] Nigel H. Goddard, "Representation and Recognition of Event Sequences," In David Touretzky, Geoffrey Hinton, and Terrence Sejnowski, editors, *Proceedings 1988 Connectionist Models Summer School*, pages 485–496, San Mateo, California, 1989. Morgan Kaufmann.

[Goddard *et al.*, 1988] Nigel H. Goddard, Kenton Lynne, and Toby Mintz, "The Rochester Connectionist Simulator," Technical Report TR233, Department of Computer Science, University of Rochester, NY 14627, March 1988.

[Gould and Shah, 1989] Kristine Gould and Mubarak Shah, "The Trajectory Primal Sketch: A Multi-Scale Scheme for Representing Motion Characteristics," In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 79–85, 1989.

[Gray and Singer, 1989] C. M. Gray and W. Singer, "Stimulus-specific neuronal oscillations in orientation specific columns of cat visual cortex," *Proceedings National Academy of Sciences*, 86:1698–1702, 1989.

[Hanson and Kegl, 1987] Steven J. Hanson and J. Kegl, "Parsnip: a connectionist network that learns natural language grammar from exposure to natural language sentences," In *Proceedings of the Ninth Annual Conference of the Cognitive Science Society*, pages 106–119, 1987.

[Hertz *et al.*, 1991] John Hertz, Anders Krogh, and Richard G. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Wesley, Redwood City, CA, 1991.

[Hildreth *et al.*, 1989] Ellen C. Hildreth, Norberto M. Grzywacz, Edward H. Adelson, and Victor K. Inada, "The Perceptual Buildup of Three-Dimensional Structure from Motion," Technical Report AI Memo 1141, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, August 1989.

[Hoenkamp, 1978] Edward Hoenkamp, "Perceptual Cues that Determine the Labelling of Human Gait," *Journal of Human Movement Studies*, 4:59–69, 1978.

[Hoffman and Flinchbaugh, 1982] D.D. Hoffman and B.E. Flinchbaugh, "The interpretation of biological motion," *Biological Cybernetics*, 42:195–204, 1982.

[Hogg, 1984] David Crossland Hogg, *Interpreting Images of a Known Moving Object*, PhD thesis, Department of Computer Science, University of Sussex, 1984.

[Honavar and Uhr, 1989] Vasant Honavar and Leonard Uhr, "A Network of Neruon-Like Units that Learns to Perceive by Generantion as well as Reweighting of its Links," In David Touretzky, Geoffrey Hinton, and Terrence Sejnowski, editors, *Proceedings 1988 Connectionist Models Summer School*, pages 472–484, San Mateo, California, 1989. Morgan Kaufmann.

[Hopfield, 1982] John J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities," *Proceedings National Academy of Science*, 79:2554, 1982.

[Huang *et al.*, 1990] X. D. Huang, Y. Ariki, and Mervyn A. Jack, *Hidden Markov models for speech recognition*, Edinburgh University Press, Edinburgh, 1990.

[Hummel and Biederman, 1990] John E. Hummel and Irving Biederman, "Dynamic Binding: A Basis for the Representation of Shape by Neural Networks," In *Proceedings Conference of the Cognitive Science Society 1990*, pages 614–621, 1990.

[Hummel *et al.*, 1989] John E. Hummel, Irving Biederman, Peter C. Gerhardstein, and H. John Hilton, "From Image Edges to Geons: A Connectionist Approach," In David Touretzky, Geoffrey Hinton, and Terrence Sejnowski, editors, *Proceedings 1988 Connectionist Models Summer School*, pages 462–471, San Mateo, California, 1989. Morgan Kaufmann.

[Huttnelocher and Ullman, 1987] D. P. Huttnelocher and S. Ullman, "Object Recognition Using Alignment," In *First International Conference on Computer Vision*, pages 102–111, 1987.

[Jain *et al.*, 1979] R. Jain, W. N. Martin, and J. K. Aggarwal, "Segmentation through the detection of changes due to motion," *Computer Graphics and Image Processing*, 11:13–34, 1979.

[Jenkin, 1986] Michael Jenkin, "Tracking Three Dimensional Moving Light Displays," In N. I. Badler and J. K. Tsotsos, editors, *Motion: Representation and Perception*. Elsevier Science Publishing, 1986.

[Johansson, 1973] Gunnar Johansson, "Visual perception of biological motion and a model for its analysis," *Perception and Psychophysics*, 14:201–211, 1973.

[Johansson, 1976] Gunnar Johansson, "Spatio-Temporal Differentiation and Integration in Visual Motion Perception," *Psychological Research*, 38:379–393, 1976.

[Jordan, 1986] Michael I. Jordan, "Serial order: A parallel distributed processing approach," Technical Report 8604, Institute for Cognitive Science, University of California, San Diego, 1986.

[Kleinfeld, 1986] David Kleinfeld, "Sequential state generation by model neural networks," *Proceedings National Academy of Sciences*, 83:9469–9473, 1986.

[Koch *et al.*, 1986] Christof Koch, Jose Marroquin, and Alan Yuille, "Analog "Neuronal" Networks in Early Vision," *Proc. National Acadamy of Science of the USA*, 83:4263–4267, June 1986.

[Koch and Ullman, 1985] Christof Koch and Shimon Ullman, "Shifts in Selective Visual Attention: towards the underlying neural circuitry," *Human Neurobiology*, 4:219–227, 1985.

[Koch *et al.*, 1989] Christof Koch, H. T. Wang, Bimal Mathur, Andrew Hsu, and Humbert Suarez, "Computing Optical Flow in Resistive Networks and in the Primate Visual System," In *Proc. IEEE Workshop on Visual Motion*, pages 62–72, Irvine, California, March 1989.

[Koffka, 1935] Kurt Koffka, *Principles of Gestalt Psychology*, Harcourt, Brace and Company, New York, 1935.

[Kosslyn, 1987] Steven M. Kosslyn, "Seeing and Imagining in the Cerebral Hemisphere: a Computational Approach," *Psychological Review*, 94(2):148–175, 1987.

[Kozlowski and Cutting, 1977] L.T. Kozlowski and J.E. Cutting, "Recognizing the sex of walker from dynamic point-light displays." *Perception and Psychophysics*, 21(6):575–580, 1977.

[Lippmann, 1989] Richard P. Lippmann, "Review of Neural Networks for Speech Recognition," *Neural Computation*, 1:1–38, 1989.

[Lisberger *et al.*, 1987] Stephen G. Lisberger, E.J. Morris, and L. Tychsen, "Visual Motion Processing and Sensory-Motor Integrationfor Smooth-Pursuit Eye Movements," *Annual Review Neuroscience*, 10:97–129, 1987.

[Lowe, 1985] David G. Lowe, *Perceptual Organization and Visual Recognition*, Kluwer Academic Publishers, Boston, Mass., 1985.

[Marr and Vaina, 1982] D. Marr and Lucia Vaina, "Representation and Recognition of the Movements of Shapes," *Proceedings of the Royal Society of London B*, 214:501–524, 1982.

[Marr, 1982] David Marr, *Vision*, W. H. Freeman and Co., San Francisco, CA, 1982.

[Maunsell and Newsome, 1987] John H. R. Maunsell and William T. Newsome, "Visual Processing in Monkey Extrastriate Cortex," *Annual Review of Neuroscience*, 10:363–401, 1987.

[Minsky, 1980] Marvin Minsky, "K-lines: A theory of memory," *Cognitive Science*, 4:117–133, 1980.

[Mishkin *et al.*, 1983] M. Mishkin, L. G. Ungerleider, and K.A. Macko, "Object Vision and Spatial Vision: Two Cortical Pathways," *Trends in Neuroscience*, 6:414–417, 1983.

[Mishkin, 1982] Mortimer Mishkin, "A Memory System in the Monkey," *Phil. Trans. Royal Soc. London B*, 298:85–95, 1982.

[Mishkin and Appenzeller, 1987] Mortimer Mishkin and Tim Appenzeller, "The Anatomy of Memory," *Scientific American*, 1987.

[Mozer, 1988] Michael C. Mozer, "The Perception of Multiple Objects: a Parallel Distributed Processing Approach," Technical Report 8803, Institute for Cognitive Science, University of California, San Diego, April 1988.

[Mozer, 1989] Michael C. Mozer, "A Focused Back-propagation Algorithm for Temporal Pattern Recognition," *Complex Systems*, 3:349–381, 1989.

[Mozer, 1991] Michael C. Mozer, *The Perception of Multiple Objects: a Connectionist Approach*, MIT Press, Cambridge, MA, 1991.

[Olson, 1989] Thomas J. Olson, *An Architectural Model of Visual Motion Understanding*, PhD thesis, University of Rochester, 1989, also Technical Report 305, Computer Science Department.

[O'Rourke and Badler, 1980] Joseph O'Rourke and Norman I. Badler, "Model Based Image Analysis of Human Motion using Constraint Propagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(6):522–536, November 1980.

[Parker, 1985] David B. Parker, "Learning Logic," Technical Report TR-47, Center for Computational Research in Economics and Management Science, Massachusetts Institute of Technology, 1985.

[Pearlmutter, 1990] Barak A. Pearlmutter, "Dynamic Recurrent Neural Networks," Technical Report CMU-CS-90-196, Department of Computer Science, Carnegie Mellon University, 1990.

[Pearlmutter, 1989] Barrak A. Pearlmutter, "Learning State Space Trajectories in Recurrent Neural Networks," *Neural Computation*, 1:263–269, 1989.

[Pineda, 1987] F. J. Pineda, "Generalization of Back-Propagation to Recurrent Neural Networks," *Physical Review Letters*, 59:2229–2232, 1987.

[Plaut, 1984] David C. Plaut, "Visual Recognition of Simple Objects by a Connection Network," Technical Report TR143, Department of Computer Science, University of Rochester, 1984.

[Poggio *et al.*, 1985] Tomaso Poggio, Vincent Torre, and Christof Koch, "Computational Vision and Regularization Theory," *Nature*, 317:314–319, September 1985.

[Poizner *et al.*, 1981] H. Poizner, U. Bellugi, and V. Lutes-Driscoll, "Perception of American Sign Language in dynamic point light displays," *Journal of Experimental Psychology: Human Perception and Performance*, 7:430–440, 1981.

[Prazdny, 1986] K. Prazdny, "Three-dimensional Structure From Long-range Apparent Motion," *Perception*, 15:619–625, 1986.

[Proffitt, 1988] Dennis R. Proffitt, "Recovering Connectivity from Moving Point-Light Displays," In Martin and Aggarwal, editors, *Motion Understanding: Robot and Human Vision*, chapter 9, pages 297–327. Boston: Kluwer, 1988.

[Rabiner and Juang, 1986] Lawrence. R. Rabiner and Biing-Hwang Juang, "An Introduction to Hidden Markov Models," *IEEE Acoustics, Speech and Signal Processing Magazine*, 3(1):4–16, 1986.

[Rashid, 1980] Richard F. Rashid, *Lights: A system for the interpretation of moving light displays*, PhD thesis, University of Rochester, 1980.

[Rimey and Brown, 1990] Raymond D. Rimey and Christopher M. Brown, "Selective Attention as Sequential Behaviour: Modeling Eye Movements with an Augmented Hidden Markov Model," Technical Report TR327, Department of Computer Science, University of Rochester, April 1990.

[Roberts, 1965] L. G. Roberts, "Machine Perception of Three-Dimensional Solids," In James.T. Tippett, David A. Berkowitz, Lewis C. Clapp, Charles J. Koester, and Alexander Vanderburgh, editors, *Optical and electro-optical processing*, pages 159–197. MIT Press, Cambridge, MA, 1965.

[Rohwer and Forrest, 1987] R. Rohwer and B. Forrest, "Training Time-Dependence in Neural Networks," In M. Caudill and C. Butler, editors, *IEEE First International Conference on Neural Networks*, volume 2, pages 701–708, San Diego, 1987.

[Rubin, 1986] John Rubin, *Categories of Visual Motion*, PhD thesis, Department of Psychology, Massachussetts Institute of Technology, 1986.

[Rumelhart *et al.*, 1986a] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Representations by Back-Propagating Errors," *Nature*, 323:553–536, 1986.

[Rumelhart *et al.*, 1986b] David E. Rumelhart, Paul Smolensky, James L. McClelland, and Geoffrey Hinton, "Schemata and Sequential Thought Processes in PDP Models," In David E. Rumelhart, James L. McClelland, and the PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 2: Psychological and Biological Models*. MIT Press, Cambridge, MA, 1986.

[Runeson and Frykholm, 1983] Sverker Runeson and Gunilla Frykholm, "Kinematic specification of dynamics as an informational basis for person-an-action perception: expectation, gender recognition, and deceptive intention," *Journal of Experimental Psychology: General*, 112(4):585–615, 1983.

[Scholz, 1989] John P. Scholz, "Reliability and Validity of the WATSMART Three-Dimensional Optoelectronic Motion Analysis System," *Physical Therapy*, 69(8):679–689, August 1989.

[Seibert and Waxman, 1989] M. Seibert and A. Waxman, "Spreading Activation Layers, Visual Saccades, and Invariant Representations for Neural Pattern Recognition Systems," *Neural Networks*, 2:9–27, 1989.

[Shastri and Ajjanagadde, 1989] Lokendra Shastri and Venkat Ajjanagadde, "A connectionist system for rule based reasoning with multi-place predicates and variables," Technical Report MS-CIS-89-06, Department of Computer and Information Science, University of Pennsylvania, 1989.

[Shastri and Ajjanagadde, 1990] Lokendra Shastri and Venkat Ajjanagadde, "From simple associations to systematic reasoning: a connectionist representation of rules, variables and dynamic bindings," Technical Report MS-CIS-90-05, Department of Computer and Information Science, University of Pennsylvania, 1990.

[Shiffrin, 1988] Richard M. Shiffrin, "Attention," In R.C. Attkinson, R.J. Herrnstein, G. Lindzey, and R.D. Luce, editors, *Steven's Handbook of Experimental Psychology, Volume 2: Learning and Cognition*, chapter 11, pages 739–811. John Wiley & Sons, New York, 1988.

[Smolensky, 1986] Paul Smolensky, "Foundations of Harmony Theory: Cognitive Dynamical Systems and the Subsymbolic Theory of Information Processing," In David E. Rumelhart, James L. McClelland, and the PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations*. MIT Press, Cambridge, MA, 1986.

[Sompolinksy and Kanter, 1986] H. Sompolinksy and I. Kanter, "Temporal association in Asymmetric Neural Networks," *Physical Review Letters*, 57(22):2861–2864, December 1986.

[Sumi, 1984] Shigemasa Sumi, "Upside-down presentation of the Johansson moving light-spot pattern," *Perception*, 13:283–286, 1984.

[Tan and Martin, 1986] Chew L. Tan and W. N. Martin, "A Distributed System for Analyzing Time-Varying Multiresolution Imagery," *Computer Vision, Graphics and Image Processing*, 36:162–174, 1986.

[Tanimoto and Klinger, 1980] S. Tanimoto and A. Klinger, *Structured Computer Vision*, Academic Press, New York, 1980.

[Tanimoto, 1989] Steven L. Tanimoto, "Special Section on Multiresolution Representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674-719, July 1989.

[Tank and Hopfield, 1987] D.W. Tank and J.J. Hopfield, "Concentrating Information in Time: Analog Neural Networks with Applications to Speech Recognition Problems," In *Proceedings First International Conference on Neural Networks 1987*, volume IV, pages 455–468, 1987.

[Tartter and Knowlton, 1981] V.C. Tartter and K. C. Knowlton, "Perceiving sign language from an array of 27 moving spots," *Nature*, 289:676–678, 1981.

[Thompson, 1971] Hunter S. Thompson, *Fear and Loathing in Las Vegas; a journey to the heart of the American Dream*, Random House, New York, 1971.

[Thompson and Whillock, 1988] William B. Thompson and Rand P. Whillock, "Occlusion-Sensitive Matching," In *Second International Conference on Computer Vision*, pages 285–289, December 1988.

[Todd *et al.*, 1988] J. T. Todd, R. A. Akerstrom, F. D. Reichel, and W. Hayes, "Apparent rotation in three-dimensional space: Effects of temporal, spatial, and structural factors," *Perception and Psychophysics*, 43:179–188, 1988.

[Todd, 1983] James T. Todd, "Perception of Gait," *Journal of Experimental Psychology: Human Perception and Performance*, 9(1):31–42, 1983.

[Todd, 1985] James T. Todd, "Perception of Structure from Motion: is Projective Correspondence of Moving Elements a Necessitary Condition?," *Journal of Experimental Psychology: Human Perception and Performance*, 11:689–710, 1985.

[Tsotsos, 1987] John K. Tsotsos, "Representational Axes and Temporal Cooperative Processes," In M. Arbib and A. Hanson, editors, *Vision, Brain and Cooperative Computation*, pages 361–418. MIT Press / Bradford Books, 1987.

[Tsotsos, 1991] John K. Tsotsos, "Localizing Stimuli in a Sensory Field using an Inhibitory Atttentional Beam," Technical Report RBCV-TR-91-37, Department of Computer Science, University of Toronto, October 1991.

[Tsotsos *et al.*, 1980] John K. Tsotsos, John Mylopoulos, H. Dominic Covvey, and Steven W. Zucker, "A Framework for Visual Motion Understanding," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(6):563–573, November 1980.

[Uhr, 1987] Leonard Uhr, editor, *Parallel Computer Vision*, Academic Press, Orlando, Florida, 1987.

[Ullman, 1979a] Shimon Ullman, "The Interpretation of Structure from Motion," *Proceedings of the Royal Society of London B*, 203:405–426, 1979.

[Ullman, 1979b] Shimon Ullman, *The Interpretation of Visual Motion*, MIT Press, Cambridge, Massachussetts, 1979.

[Ullman, 1984] Shimon Ullman, "Maximizing rigidity: the incremental recovery of 3-D structure from rigid and non-rigid motion," *Perception*, 13:255–274, 1984.

[von der Malsburg, 1987] Christof von der Malsburg, "Synaptic plasticity as a basis of brain-organization," In J. P. Chaneaux and M. Konishi, editors, *The Neural and Molecular Bases of Learning*, pages 411–432. New York: Wiley, 1987.

[Waibel, 1989] Alex Waibel, "Modular Construction of Time-Delay Neural Networks for Speech Recognition," *Neural Computation*, 1:39–46, 1989.

[Wallach and O'Connell, 1953] Hans Wallach and D. N. O'Connell, "The Kinetic Depth Effect," *The Journal of Experimental Psychology*, 45:205–217, 1953.

[Webb and Aggarwal, 1982] Jon A. Webb and J. K. Aggarwal, "Structure from Motion of Rigid and Jointed Objects," *Artificial Intelligence*, 19:107–130, 1982.

[Weinshall, 1988] Daphna Weinshall, "Application of Qualitiative Depth and Shape from Stereo," In *Second International Conference on Computer Vision*, pages 144–145, December 1988.

[Werbos, 1974] P.J. Werbos, *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*, PhD thesis, Harvard University, 1974.

[Whitehead and Ballard, 1990] Steven D. Whitehead and Dana H. Ballard, "Active Perception and Reinforcement Learning," Technical Report TR331, Department of Computer Science, University of Rochester, February 1990.

[Williams and Zipser, 1989] R. J. Williams and D. Zipser, "A Learning Algorithm for Continually Running Fully Recurrent Neural Networks," *Connection Science*, 1:270–280, 1989.

[Yarbus, 1967] A. L. Yarbus, *Eye Movements and Vision*, Plenum Press, New York, 1967.

[Zemel *et al.*, 1990] Richard S. Zemel, Michael C. Mozer, and Geoffrey E. Hinton, "TRAFFIC: Recognizing Objects Using Hierarchical Reference Frame Transformations," In David S. Touretzky, editor, *Advances in Neural Information Processing 2*, pages 266–273. Morgan Kaufmann, San Mateo, CA, 1990.

# A Scenarios for Walking, Running and Skipping

Here we show the scenario models used in the main experiments reported in Chapter 6. There are three models, for walking, skipping and running to the right. The leftward versions are the mirror image of these. Each model consists of the specification of events for the legs scenario followed by that for the arms. An event is specified by its temporal parameters followed by the limb configurations that enable and support it.

There are five temporal parameters, the unit being the input frame (60/sec). The first three indicate minimum, average and maximum duration of the event. The last two indicate the earliest and latest time of occurrence of the event with respect to an arbitrary baseline set at the time of occurrence of the first legs event. An example is:

```
Event D (6, 8, 9, 24, 27)
```

which means the minimum, average and maximum durations of event D are 6, 8 and 9 frames respectively (or 30, 40 and 45 simulation steps), and the earliest and latest frames at which event D occurs with respect to the time that leg event A occurs are frame 24 and frame 27 respectively (or simulation steps 120 and 135). These parameters are used in the interval and delay units.

The format of a pair-of-limbs configuration is best explained by example. In a legs event, "fp3+/f-3-" means that the proximal leg is configured "fp3+" and the distal leg "f-3-". The four characters specifying one limb configuration represent, in order: local segment (e.g., thigh) orientation; local segment angular velocity; remote segment (e.g., calf) orientation; and remote segment angular velocity. The letters "a-l" indicate the 12 divisions of orientation used for local segments (for specifying remote regions), measured clockwise from the vertical. The letters "Mm-+pP" indicate the 6 divisions of angular velocity used for local segments ($\leq -200$, -200 to -100, -100 to 0, 0 to 100, 100 to 200, $\geq 200$ degrees/sec, clockwise being positive). The numbers "1-4" indicate quadrants of orientation used for the remote segments. For remote segments, angular velocities are coded as positive or negative with no further refinement.

## A.1 Walking Model

```
Legs:
 Event A (5, 7, 8, 0, 0)
  enabled by:   gm3+/f+2+ gm3+/e+2+ gm3+/f+3+;
  supported by: gm3+/f+2+ gm3+/fp2+ gM3+/fp2+ fM3+/fp3+
      gM3+/f+3+ gM3+/fp3+ fM3+/fP3+ gm3+/fp3+ gm3+/e+2+
      gM3+/ep2+ gm3+/f+3+ gM3+/fP3+ gM3+/f+2+;
 Event B (4, 6, 7, 5, 8)
  enabled by:   fM3-/fp3+ fM3-/fP3+;
  supported by: fM3-/fp3+ fM3-/fP3+;
 Event C (10, 13, 14, 11, 14)
  enabled by:   fm3-/fp3+ fm3-/gP3+ fm3-/fP3+;
  supported by: f-3-/g+3+ f-3-/gp3+ f+3-/g+3+ fm3-/fp3+
      f-3-/fp3+ f+3-/gp3+ f+2-/gp3+ f+2-/g+3+ fm3-/gP3+
      f+3-/fp3+ f-2-/g+3+ f-3-/fP3+ f-2-/gp3+ e-2-/g+3+
      fm3-/fP3+ e-2-/gp3+ f-2-/gP3+;
 Event D (6, 8, 9, 24, 27)
  enabled by:   f+2+/g+3+ f-2+/g+3+ e-2+/g+3+ f-2+/gp3+;
  supported by: f+2+/g+3+ f+2+/g-3+ f+2+/gm3+ f+3+/gM3+
      f-2+/g+3+ f-2+/g-3+ e-2+/g+3+ e-2+/g-3+ e+2+/gm3+
      f-2+/gp3+ f+3+/gm3+ fm2+/gp3+ fm2+/g+3+ f+2+/gM3+;
 Event E (18, 21, 24, 31, 35)
  enabled by:   fp2+/gm3+ fp3+/fM3+ fp3+/gM3+ ep2+/gM3+;
  supported by: g+3+/f-3- g+3+/f+3- gp3+/f-3- g+3+/f+2-
      fp2+/gM3+ fp3+/fM3+ fp3+/fM3- fp3+/fm3- fp3+/f-3-
      gp3+/f+3- gp3+/f+2- gP3+/fm3- fp3+/gM3+ fp3+/f+3-
      fP3+/fM3+ fP3+/f-3- gp3+/f-2- fp3+/gm3+ fP3+/fM3-
      fP3+/fm3- gp3+/e-2- fP3+/gM3+ gP3+/f-2- gp3+/f-2+
      gp3+/fm2+;
 Event F (4, 6, 9, 51, 56)
  enabled by:   g+3+/f+2- g+3+/f+2+ g+3+/f-2- g+3+/f-2+
      g+3+/e-2- g+3+/fm2+;
  supported by: g+3+/f+2- g+3+/f+2+ g-3+/f+2+ g+3+/f-2-
      g+3+/f-2+ g-3+/f-2+ g+3+/e-2+ g-3+/e-2+;.

Arms:
 Event A (3, 10, 20, 42, 53)
  enabled by:   f-2-/g+2+ f-2-/gp2+ f-2-/g+3+ f-2-/f+2+
      f-2-/gp3+;
  supported by: f-2-/g-3+ f+2+/gm2- f+2+/g-2- f+2-/gp3+
      f+2-/g+3+ f+2+/g+3+ f+2+/g-3+ f-2-/g+2+ f-2-/gp2+
```

```
     f-2+/g+2- f+2+/g+2- f-2-/g+3+ e+2+/g-3- f+2+/g-3-
     f+2-/g+2+ f+2+/g+3- f-2-/gp3+ f+2+/gm3- e-2-/g+3+
     e+2-/g+3+ f+2-/g-3+ f+2+/gM2-;
Event B (13, 21, 30, -6, 12)
 enabled by:   fp2+/g-2- fp2+/gm2- fp2+/gm3- fp2+/g-3-
     fp2+/g+3+;
 supported by: gP2+/gm2- gp2+/gm2- g+2+/gm2- g+2+/fm2-
     fp2+/g-2- fp2+/gm2- f+2+/gm2- f+2+/g-2- fp2+/fm2-
     f+2+/fm2- g+2+/f-2- gp2+/f-2- gp3+/f+2- f+2+/g-3-
     fp2+/gm3- fP2+/gm2- fP2+/fm2- fP2+/fM2- gP2+/fM2-
     gp2+/fM2- gp2+/fm2- fp2+/fM2- gp2+/gM2- g+2+/gM2-
     fp2+/g-3- gp3+/f-2- gP2+/gM2- f+2+/gm3- fP2+/gM3-
     gP3+/fM2- gp3+/fm2- fP2+/gm3- fP2+/gM2- fp2+/gM2-
     gP3+/fm2- gP2+/g-2- gP2+/f-2- gP2+/fm2- f+2+/gM2-
     fp2+/g+3+ fp2+/g+3- fP2+/g-3- fp2+/g-3+;
Event C (4, 11, 22, 13, 26)
 enabled by:   g+2+/fm2- f+2+/fm2- g+2+/f-2- g+3+/f-2-
      g+3+/fp2+;

 supported by: g+2+/fm2- g-3+/f-2- g+3+/f+2- g+3+/f+2+
     g-3+/f+2+ g+2+/f-2- g+2-/f-2+ g+2-/f+2+ g-2-/f+2+
     g-2-/fp2+ g+3+/fm2- g+3+/f-2- g-3-/e+2+ g-3-/f+2+
     g+2+/f+2- f+2+/f-2- g+3-/f+2+ g-3-/fp2+ g+3+/e-2-
     g+3+/e+2- g-3+/f+2- g+3+/fp2+ g+3-/fp2+ g-3-/fP2+
     g-3+/fp2+;
Event D (9, 18, 24, 27, 37)
 enabled by:   gm2-/fp2+ gm2-/f+2+ fm2-/fp2+ gm3-/fp2+
     gm3-/f+2+ gm3-/fP2+;
 supported by: gm2-/gP2+ gm2-/gp2+ gm2-/g+2+ fm2-/g+2+
     g-2-/f+2+ g-2-/fp2+ gm2-/fp2+ gm2-/f+2+ fm2-/fp2+
     fm2-/f+2+ f-2-/gp2+ gm3-/fp2+ gm2-/fP2+ fm2-/fP2+
     fM2-/fP2+ fM2-/gP2+ fM2-/gp2+ fm2-/gp2+ fm2-/g+3+
     fM2-/fp2+ gM2-/gp2+ gM2-/g+2+ gm3-/f+2+ gM3-/fP2+
     fM2-/gP3+ fm2-/gp3+ gM2-/gP2+ gm3-/fP2+ gM2-/fP2+
     gM2-/fp2+ fm2-/gP3+ g-2-/gP2+ f-2-/gP2+ gM2-/f+2+
     fm2-/gP2+;.
```

## A.2 Running Model

```
Legs:
 Event A (7, 9, 12, 0, 0)
  enabled by: gm3+/fp2+ gm3+/fP2- gm3+/fm2+ gm3+/fm3+
      gm3+/f-3+ gm3+/f-2+ gm3+/f-2-;
  supported by: gm3+/fp2+ gM3+/f+2+ gM3+/f+3+ fM3+/f+3+
      fM3+/f-3+ gM3+/fp2+ gm3+/fP2+ gm3+/fm2+ gm3+/fM2+
      gM3+/fM3+ gM3+/fm3+ fM3+/fp3+ gm3+/fm3+ gM3+/f-3+
      fm3+/fm2+ fm3+/fm3+ fM3+/fm3+ gm3+/f-3+ gm3+/f-2+
      gM3+/fm2+;
 Event B (6, 8, 10, 7, 12)
  enabled by:   fM3-/f+3+ fM3-/fp3+ fM3-/fP3+ gM3-/f-3+;
  supported by: fM3-/f+3+ fM3-/fp3+ fM3-/fP3+ eM3-/fP3+
      em3-/fP3+ fm3-/fP3+ fm3-/fP3-;
 Event C (11, 12, 15, 14, 20)
  enabled by: e-3-/fP3+ f-3-/fP3+ f-3-/fP3- f+3-/gP3+
      f-3-/gP3+;
  supported by: e+3-/fP3+ ep3-/gP3+ fP3-/gP3+ fP3-/gp3+
      fP2-/g+3+ fP2-/g-3+ eP3-/gP3+ fP2-/gm3+ fP2-/gp3+
      f-2-/g+3+ f+2-/g+3+ f-3-/fP3+ f-3-/gP3- f+3-/gP3+
      fp3-/gP3+ fp3-/gp3+ f+3-/g+3+ f-2-/g-3+ fm2-/g-3+
      f+2-/g-3+ f-3-/gP3+ f+3-/g-3+ f+3-/fP3+ fp3-/g+3+
      f+3-/gp3+ f-3-/g-3+ fp2-/g-3+ fp2-/gp3+ f-2-/gm3+;
 Event D (9, 11, 14, 28, 31)
  enabled by: fp2+/gm3+ fP2+/gm3+ fm2+/g-3+ f-3+/g-3+
      f+3+/g-3+ f-2+/gm3+;
  supported by: fp2+/gm3+ f+2+/gM3+ f+3+/gM3+ f+3+/fM3+
      f-3+/fM3+ f+3+/fM3- fp3+/fM3- fp2+/gM3+ fm2+/g-3+
      fm2+/gm3+ fM2+/gm3+ fM3+/gM3+ fm3+/gM3+ fp3+/fM3+
      fm3+/gm3+ f-3+/gM3+ fm3+/fm3+ fm2+/fm3+ fm3+/fM3+
      f-3+/gm3+ f-2+/gm3+ f-3+/gM3- fm2+/gM3+;
 Event E (12, 14, 16, 37, 45)
  enabled by:   fP3+/fM3-;
  supported by: fP3+/fM3- fP3+/eM3- fP3+/em3- fP3+/e-3-
      fP3+/e+3- gP3+/ep3- gP3+/fP3- gp3+/fP3- gP3+/eP3-
      gp3+/fP2- fP3+/f-3- fP3+/fm3- fP3-/fm3- fP3-/f-3-
      gP3-/f-3- gP3+/f+3- gP3+/fp3- gp3+/fp3- gP3+/f-3-
      fP3+/f+3- gp3+/f+3- gp3+/fp2-;
 Event F (2, 5, 9, 51, 58)
  enabled by:   g+3+/fP2- g+3+/f+2- g+3+/f+3- g+3+/fp3-;
  supported by: g+3+/fP2- g-3+/fP2- g+3+/f-2- g+3+/f+2-
```

```
      g+3+/f+3- g-3+/f-2- g-3+/fm2- g-3+/fm2+ g-3+/f+2-
      g-3+/f+3- g+3+/fp3- g-3+/f-3- g-3+/f-3+ g-3+/f+3+
      g-3+/fp2-;.

Arms:
 Event A (3, 7, 14, 16, 32)
  enabled by: g+2-/f+1+ g+2-/f+2+ g+2-/fm1+ h+2-/g+2+
      h+2-/gp2+ g+2+/g-2- g+2+/gm2- g+2-/gp1+ g+2-/fm2-
      h+2-/f+2- h+2+/f-2-;
  supported by: g+2-/f+2+ g-2-/f+2+ g-2-/fp2+ g+2-/fm1+
      g+2-/f-1+ g-2-/f+1+ h-2-/g+2+ h+2-/gp2+ h-2-/gp2+
      g+2+/g+2- g+2+/g-2- g+2-/g+1+ g+2-/gp1+ g+2-/g+2+
      g+2-/g-2+ g-2-/g+2+ g+2-/g+2- g+3+/g-2- g+3+/g+2-
      g+3-/g+2- g+3-/g+2+ g+2-/f-2- g-2-/f-2- h+2-/f+2-
      h-2-/f+2+ h-2-/fp2+ h+2-/f+2+;
 Event B (11, 18, 28, 25, 36)
  enabled by: gm2-/fp2+ hm2-/g+2+ gm2-/g+2+ gm2-/gp2+
      gm2-/f+2+ hm2-/fp2+;
  supported by: g-2-/gP2+ gm2-/gP2+ gM2-/gP2+ gm2-/fp2+
      gM2-/fp2+ gM2-/fP2+ gM1-/gP2+ fM1-/gP2+ fM1-/gP2-
      fM1+/gP2- fm1+/gP2- gM2-/f+2+ gM2-/g+2+ gM2-/gp2+
      gM1-/gP2- fm1-/gP2- fm1+/gp2- fm1+/g+2- hM2-/g+2+
      gM2-/g+3+ gM2-/gp3+ gm1-/gp3+ gm2-/gp3+ gm2-/g+2+
      gM1-/gp2+ gM1-/gp3+ gm2-/gp2+ g-2-/gp2+ gm2-/f+2+
      fm2-/gp2+ fm2-/g+2- hm2-/f+2+ f-2-/gP2+ fm2-/gP2+;
 Event C (4, 10, 19, -18, 0)
  enabled by: g-2-/gP2+ f-1+/gp2- f-1+/gP2- f-1+/g+2-
      g-1-/gp3- g-2-/gp3+ g-1-/gp3+ g-2-/g+2+ g-2-/gp2+
      f-2-/g+2- f-2-/hp2+;
  supported by: g-2-/gP2+ f-1+/gp2- f+1+/g+2- f+2+/g+2-
      f+2+/g-2- f+1+/gp2- f+2+/gp2- g+2+/gM2- f-1+/g+2-
      f+1+/g-2- g-1-/gp3- g+1-/gp3- g+1-/gP3- g+2+/hp2-
      g+2+/h-2- g-2-/gp3- g+1+/gp3- g+1+/gp2- g+2+/gp2-
      g+1-/g+3- g+1+/g+2- g+2+/g+2- g+2+/g-2- g+2+/gm2-
      g+2-/g+2+ g-2-/g+2+ g+2-/g+2- g-2-/gp2+ g-2-/g+3+
      g+2-/g+3+ g+2-/g+3- g+2+/g+3- f-2-/g-2- f+2+/gm2-
      f-2-/hp2+ f-2-/hp2- f+2-/h+2- f+2+/h-2- g+2-/gP2+
      f-2-/gp2+ f-2-/h+2+ f+2+/h+2-;
 Event D (14, 25, 39, -14, 8)
  enabled by: fp2+/g-2- fp2+/gm2- gp2+/gM2- gp1+/gP3-
      gp1+/gp2- gp2+/h+2- gp2+/gm2- fp2+/h-2-;
  supported by: gP2+/g-2- gP2+/gm2- gP2+/gM2- fp2+/g-2-
```

```
fp2+/gm2- fp2+/gM2- fP2+/gM2- gP2+/gM1- gP2+/fM1-
gP2-/fM1- gP2-/fM1+ gP2-/fm1+ gp2-/f-1+ f+2+/gM2-
g+2+/gM2- gp2+/gM2- gP2-/gM1- gP2-/fm1- gP2-/f-1+
gp2-/f+1+ gp2-/f+2+ gp2-/fm1+ gp1+/gp2- gp1+/gP2-
g+1+/gP2- g+2+/hP2- g+2+/hp2- g+2+/h+2- g+2+/h-2-
g+2+/hm2- g+2+/hM2- g+3+/gM2- gp3+/gM2- gp3+/gm1-
gp3-/g-1- gp3-/g+1- gP3-/g+1- gP3-/gp1+ gp2-/gp1+
gP2-/gp1+ gP2-/g+1+ hP2-/g+2+ hp2-/g+2+ gp2+/h+2-
gp2+/h-2- gp3+/gm2- gp3+/g-2- gp3-/g-2- gp3-/g+1+
gp2-/g+1+ gp2-/g+2+ g+1+/g+2- gp1+/g+2- g+2+/g+2-
g-2+/g+2- g+2+/g-2- g+2+/gm2- gp2+/gM1- gp3+/gM1-
gp3+/g-1- g+3-/g+1- g+2-/g+1+ gp2+/gm2- gp2+/g-2-
f+2+/gm2- gp2+/fm2- fp2+/hm2- f+2+/hm2- gP2+/f-2-
gP2+/fm2- hp2+/f-2- hp2-/f-2- gP2+/g+2- gp2+/f-2-;.
```

## A.3 Skipping Model

```
Legs:
 Event A (5, 6, 8, 0, 0)
  enabled by:   gm3+/f+2+ gm3+/fp3+ gm3+/f+3+ gm3+/f-2+;
  supported by: fM3+/fp3+ gM3+/f+3+ gM3+/fp3+ fM3+/fP3+
      gM3+/fP3+ gM3+/f-2+;
 Event B (4, 6, 9, 5, 8)
  enabled by:   fM3-/fP3+ fM3-/fP3-;
  supported by: fM3-/fp3+ fM3-/fP3+ fM3-/f+3+ fM3-/fP3-
      eM3-/fp3- eM3-/fp3+ eM3-/f+3+ eM3-/f+3- eM3-/fP3-
      eM3-/gp3- eM3-/g+3-;
 Event C (10, 14, 16, 11, 14)
  enabled by: fm3-/fP3+ em3-/g+3+ em3-/f+3- em3-/f+3+
      em3-/fp3-;
  supported by: f-3-/g+3+ fm3-/gp3+ f+3-/g+3+ f+2-/gp3+
      f+2-/g+3+ fp2-/gp3+ em3-/f+3- e-3-/f+3- e+3-/f-3-
      e+3-/f+3+ ep3-/f+3+ ep2-/f+3+ ep2-/fp3+ f+2-/gP3+
      e-3-/f-3+ e-3-/f-3- e+3-/f+3- f+2-/f+3+ f+2-/fp3+
      f-3-/g+3- f+3-/g+3- e+2-/f-3+ e+2-/f+3+ e+2-/fp3+
      e+2-/fP3+ e+3-/f-3+ e+2-/gP3+ e-3-/g+3+ e+3-/g+3-
      ep3-/g+3- ep3-/g+3+ ep3-/g-3+ eP3-/g-3+ e-3-/g+3-
      e+3-/g+3+ eP3-/gp3+ eP3-/g+3+ fP3-/g+3+ em3-/gp3-
      ep3-/gp3+;
 Event D (5, 7, 9, 24, 28)
  enabled by: f-2+/gp3+ fp3+/g-3+ f+2+/gP3+ e+2+/fP3+
      e+2+/gP3+ eP3+/g-3+ fP3+/g+3+;
  supported by: f+2+/g+3+ f+2+/g-3+ f+2+/gm3+ fp3+/fM3+
      f+3+/gM3+ fp3+/gM3+ f-2+/g+3+ f-2+/g-3+ fp3+/gm3+
      f+3+/gm3+ f+3+/g-3+ f-2+/gm3+ fp3+/g-3+ f+2+/gP3+
      f+2+/gp3+ f-2+/gM3+ ep2+/gP3+ ep2+/gp3+ ep3+/g+3+
      e+2+/gP3+ e+2+/gp3+ fP3+/g-3+;
 Event E (8, 9, 11, 33, 34)
  enabled by:   fP3+/fM3+ fP3+/gM3+;
  supported by: gp3+/fm3- fp3+/fM3- fP3+/fM3+ fP3+/fM3-
      fP3+/fm3- fP3-/fM3- fp3-/eM3- fp3+/eM3- fP3-/eM3-
      gp3-/eM3- fp3-/em3- gp3-/em3-;
 Event F (16, 18, 19, 41, 44)
  enabled by: g+3+/f-3- f+3+/fM3- f+3-/em3- f+3+/em3-
      g+3-/eM3- g+3+/e-3- g+3-/e-3-;
  supported by: g+3+/f+3- g+3+/f+2- g+3+/f+2+ g-3+/f+2+
      gp3+/f+2- g+3+/f-2+ g-3+/f-2+ gp3+/f-2+ f+3+/fM3-
```

g-3+/f+3+ g-3+/fp3+ gp3+/fp2- g+3+/em3- f+3-/em3-
f+3-/e-3- f-3-/e+3- f+3+/e+3- f+3+/ep3- f+3+/ep2-
fp3+/ep2- gP3+/f+2- gP3+/f+2+ gp3+/f+2+ f-3+/e-3-
f-3-/e-3- f+3-/e+3- f+3+/f+2- fp3+/f+2- g+3-/f-3-
g+3-/f+3- f+3+/eM3- f+3-/eM3- f-3+/e+2- f+3+/e+2-
fp3+/e+2- fP3+/e+2- fP3+/e+2+ gP3+/ep2+ gp3+/ep2+
g+3+/ep3+ f-3+/e+3- gP3+/e+2- gP3+/e+2+ gp3+/e+2+
g+3+/e-3- g+3-/e+3- g+3-/ep3- g+3+/ep3- g-3+/ep3-
g-3+/eP3- g-3+/eP3+ g-3+/fP3+ g+3+/e+3- gp3+/eP3-
g+3+/eP3- g+3+/fP3- gp3+/ep3- g+3+/fP3+;.

Arms:
Event A (3, 8, 17, 9, 21)
enabled by: g+2+/fm2- g+3+/f+2- g+3+/fm2- g+2+/f+2-h
+3+/fm2- g+3+/e+2- g+2-/e-2-;
supported by: g+2+/fm2- g-2+/fm2- g-3+/f-2- g-3-/f+2-
g+3+/f+2- g-3+/f+2+ g+2-/f+2+ g-2-/f+2+ g+3+/fm2-
g+3+/f-2- g-3-/e+2+ g-3-/f+2+ g+3+/e+2- g-2+/f-2-
g+2-/f-2- h-3-/fp2+ g-2-/fm2- g-2-/e-2- g+2-/e-2-
g+2-/f+2- h+3+/e-2- h+3+/em2- h-3-/e-2- h-3-/e+2-;
Event B (13, 21, 27, 21, 26)
enabled by: gm2-/fp2+ gm3-/f+2+ gm3-/f-2+ gm3-/e+2+
gm2-/e+2- hm3-/fp2+;
supported by: gm2-/gp2+ gm2-/g+2+ fm2-/g+2+ fm2-/g-2+
gm2-/fp2+ gm2-/f+2+ fm2-/fp2+ gm3-/fp2+ gm2-/fP2+
fm2-/fP2+ fM2-/fP2+ fM2-/gP2+ fm2-/gp2+ fm2-/g+3+
fM2-/fp2+ gM2-/gp2+ gm3-/f+2+ gM3-/fP2+ fM2-/gP3+
fM2-/gp3+ fm2-/gp3+ gM2-/gP2+ fm2-/h+3+ gm3-/fP2+
gM2-/fP2+ gM2-/fp2+ fm2-/gP3+ gM2-/f+2+ fm2-/gP2+
gM3-/f+2+ gm3-/e+2+ gm2-/e+2- gm2-/e+2+ g-2-/ep2+
g-2-/fP2+ fm2-/g-2- gm2-/ep2+ gm2+/eP2+ fm2+/eP2+
fm2-/eP2+ em2-/fP2+ em2-/fp2+ em2-/gp2+ gm2-/f-2+
fM2-/hp3+ fm2-/hp3+ em2-/h+3+;
Event C (4, 12, 27, 37, 51)
enabled by: f-2-/gp2+ f-2-/g+3+ f-2-/gp3+ f-2-/g-2+
e-2-/gP2+ e-2-/gp2+ e-2-/g-2- e-2-/h+3+;
supported by: f-2-/g-3+ f+2-/g-3- g+2+/gm2- f+2+/gm2-
f+2+/g-2- f+2-/gp3+ f+2-/g+3+ f+2+/g-3+ f-2-/gp2+
f+2+/g+2- f-2-/g+3+ e+2+/g-3- f+2+/g-3- f+2-/g+2+
f-2-/gp3+ f+2+/gm3- e+2-/g+3+ e-2-/gp3+ f-2-/gP3+
f+2+/gM2- f-2+/gm3- f+2+/gM3- f-2-/g-2+ f-2-/g+2-
e-2-/gp2+ e+2+/gm3- e-2-/g-2- e+2-/gm2- e+2+/gm2-

```
     e-2-/g+2- f-2+/gm2- f+2-/g+2- e-2-/h-3- e+2-/h-3-;
Event D (9, 19, 24, -6, 4)
 enabled by: fp2+/gm2- fp2+/gm3- gp2+/fm2- fp2+/gM2-
     fp2+/h-3- ep2+/g-2- ep2+/gm2-;
 supported by: gp2+/gm2- g+2+/fm2- fp2+/gm2- fp2+/fm2-
     gp2+/f-2- gp3+/f+2- fp2+/gm3- fP2+/gm2- fP2+/fm2-
     fP2+/fM2- gP2+/fM2- gp2+/fm2- fp2+/fM2- gp2+/gM2-
     gp3+/f-2- gP2+/gM2- fP2+/gM3- gP3+/fM2- gp3+/fM2-
     gp3+/fm2- fP2+/gm3- fP2+/gM2- gP3+/fm2- gp3+/e-2-
     gP3+/f-2- gP2+/fm2- gP2+/e-2- gp2+/e-2- fP2+/g-2-
     eP2+/gm2+ eP2+/fm2+ eP2+/fm2- fP2+/em2- fp2+/em2-
     gp2+/em2- fp2+/hm3- hp3+/fM2- hp3+/fm2-;.
```